НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

"На правах рукопису"
УДК__004.457_____

«До захисту допущено»
Завідувач кафедри
_____ О.В. Коваль
(підпис) (ініціали, прізвище)
“___”_____2018р.

Магістерська дисертація

зі спеціальності - 121 Інженерія програмного забезпечення
за спеціалізацією - Програмне забезпечення веб - технологій та мобільних пристроїв
на тему ІНСТРУМЕНТАЛЬНІ ЗАСОБИ АВТОМАТИЗОВАННОГО НАЛАШТУВАННЯ ДИНАМІЧНИХ РЕЄСТРІВ ЕЛЕКТРОННИХ ІНФОРМАЦІЙНИХ РЕСУРСІВ _____

Виконав (-ла): студент (-ка) _6_курсу, групи_ТІ-71мп
_Чайка Антон Юрійович_____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник _доцент, к.ф-м.н. Карпенко С.Г.__ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ - 2018

Національний технічний університет України
“ Київський політехнічний інститут ім. Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти другий, магістерський

зі спеціальності - 121 Інженерія програмного забезпечення
за спеціалізацією - Програмне забезпечення веб - технологій та мобільних пристроїв

ЗАТВЕРДЖУЮ
Завідувач кафедри
Коваль О.В. _____
(прізвище, ініціали) (підпис)
«____» _____ 2018р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Чайка Антон Юрійович
(прізвище, ім’я, по батькові)

1. Тема дисертації ІНСТРУМЕНТАЛЬНІ ЗАСОБИ АВТОМАТИЗОВАННОГО НАЛАШТУВАННЯ ДИНАМІЧНИХ РЕЄСТРІВ ЕЛЕКТРОННИХ ІНФОРМАЦІЙНИХ РЕСУРСІВ

Науковий керівник к.ф-м.н., доцент_ Карпенко С.Г.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “___” _____ 2018 року №___

2. Строк подання студентом дисертації “ ” 2018 року

3. Об’єкт дослідження динамічні реєстри інформації

4. Предмет дослідження _ інструментальні засоби та системи керування електронними інформаційними ресурсами .

5. Перелік питань, які потрібно розробити
1) проаналізувати визначення динамічних реєстрів;
2) проаналізувати сучасні проблеми та способи вирішення накопичення інформації;
3) спроектувати систему зберігання електронних інформаційних ресурсів;
4) розробити програмне забезпечення.

6. Орієнтований перелік ілюстративного матеріалу
мета, постановка задачі, аналіз динамічних реєстрів, архітектура системи, інтерфейс користувача, результати дослідження

7. Орієнтований перелік публікацій _____ Проблеми автоматичного налаштування динамічних реєстрів електронних інформаційних ресурсів // Карпенко С.Г., Чайка А.Ю. // Сучасні проблеми наукового забезпечення енергетики: Матеріали XVI Міжнародної науково-практичної конференції аспірантів, магістрів і студентів, м. Київ, 24-27 квітня 2018 р. У К.: КПІ ім. Ігоря Сікорського», 2018. – С. 148._

8. Дата видачі завдання «__» _____ ____р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	30.09.17р.	
2	Збір інформації	02.10.17р. – 18.11.17р.	
3	Аналіз вимог завдання, вибір методів і засобів розв'язання поставленої задачі	19.11.17р. – 31.01.18р.	
4	Підготовка матеріалів магістерської роботи	05.02.18р. – 11.05.18р.	
5	Проміжний контроль підготовки	02.04.18р. – 06.04.18р.	
6	Підготовка публікацій	05.02.17р. – 31.05.18р.	
7	Написання основних розділів автореферату	12.02.18р. – 25.05.18р.	
8	Звіт за перший рік роботи над магістерською дисертацією	31.05.18р.	

Студент _____ __Чайка А.Ю._____
(підпис) (прізвище та ініціали)

Науковий керівник _____ ___Карпенко С.Г._____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Структура й обсяг дипломної роботи. Магістерська дисертація складається зі вступу, п’яти розділів, висновку, переліку посилань з 33 найменувань, 2 додатки, і містить 9 рисунків, 23 таблиці. Повний обсяг магістерської дисертації складає 105 сторінок, з яких перелік посилань займає 2 сторінки, додатки – 6 сторінок.

Актуальність теми. Кількість інформації що генерує людина зростає із року в рік в геометричній прогресії, що призводить до проблем зберігання та пошуку необхідних даних. Пропорційно обсягам інформації зростає складність задачі обробки. Без використання спеціальних підходів та інструментів керування даними користування сучасними джерелами інформації стає практично неможливим. Для вирішення цієї проблеми використовуються інструментальні засоби автоматизованого налаштування динамічних реєстрів електронних інформаційних ресурсів.

Мета дослідження полягає у розробкці інструментальних засобів автоматичного налаштування динамічних реєстрів електронних інформаційних ресурсів.

Об’єктом дослідження є програмне забезпечення динамічних реєстрів інформації.

Предметом дослідження є інструментальні засоби та системи керування електронними інформаційними ресурсами.

Наукова новизна одержаних результатів. Найбільш суттєвими науковими результатами магістерської дисертації є:

- набуло подальшого розвитку застосування клієнт-серверного рішення для динамічного реєстру інформації з використанням технології REST для передачі повідомлень між серверною та клієнтською частиною.

Практичне значення. Розроблені інструментальні засоби автоматизованого налаштування динамічних реєстрів електронних інформаційних ресурсів забезпечують

достатньо повне і цілісне рішення задач збереження та керуванням динамічними реєстрами інформації.

Апробація результатів. Основні положення роботи доповідались та обговорювались на:

1. XVI Міжнародної науково-практичної конференції аспірантів, магістрантів і студентів (м. Київ, 24-27 квітня 2018 року);

Публікації. Наукові положення дипломної роботи опубліковані у 1й роботі

Основні публікації по темі дисертації

1. Проблеми автоматичного налаштування динамічних реєстрів електронних інформаційних ресурсів // Карпенко С.Г., Чайка А.Ю. // Сучасні проблеми наукового забезпечення енергетики: Матеріали XVI Міжнародної науково-практичної конференції аспірантів, магістрів і студентів, м. Київ, 24-27 квітня 2018 р. У К.: КПІ ім. Ігоря Сікорського», 2018. – С. 148.

Ключові слова: *ДИНАМІЧНІ РЕЄСТРИ, ІНФОРМАЦІЙНІ РЕСУРСИ*

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ І ПОЗНАЧЕНЬ

API – Application Programming Interface, сукупність засобів та правил, що вможливлюють взаємодію між окремими складниками програмного забезпечення

СКДБ – Система керування базами даних

REST – Representational State Transfer, підхід до архітектури мережевих протоколів

Фреймворк - інфраструктура програмних рішень, що полегшує розробку складних систем

ВСТУП

Через розвиток технологій, поширилось використання комп’ютерів не лише у бізнесі та наукових, інженерних задач, а і в повсякденному користуванні. Тому, найбільш популярною стала електронна форма представлення інформації — це така форма, в якій представлення користувачеві, збереження та обробка даних здійснюється за допомогою засобів електронно-обчислювальної техніки. Усі застосування визначення "електронні" ("е-") можна узагальнити за такими ознаками, як необхідність програмних та апаратних засобів для її сприйняття людиною, подання інформації в цифровому вигляді, необхідність телекомунікаційних засобів для отримання або розповсюдження інформації.

Головною характерною особливістю електронних документів є метадані. Вони стосуються інформації про дату створення та редагування, контекст, структуру та визначення елементів; юридичних, ділових, організаційних, процедурних доказів цілісності та автентичності авторів, користувачів і дослідників. Метадані є обов'язковим елементом процесу зберігання електронного документа.

Електронні ресурси — це нові специфічні об'єкти бібліотечного опрацювання, які відрізняються від інших об'єктів каталогізації, насамперед — документів на паперових носіях, що опрацьовуються в технологічних підрозділах різноманітних установ. Нова природа електронних ресурсів породила й нові джерела відомостей для опису. Режими доступу, типи носіїв, динаміка інформаційного вмісту, системні вимоги та специфіка взаємодії з користувачем — це ті нові, особливі характеристики, які зумовлюють нові можливості та умови для каталогізації електронних ресурсів та формування їх бібліографічного опису [1].

Необхідність створення правил каталогізації електронних ресурсів, бібліотечне опрацювання яких сприятиме якісному удосконаленню інформаційно-бібліотечних сервісів, зумовлена збільшенням виробництва інформації в електронному вигляді та швидким розвитком інформаційних технологій.

Більш того, дедалі ширше використання електронних документів вимагає розробки практичної системи їх класифікації - такої, що забезпечуватиме адекватну і

зрозумілу передачу змісту документів з метою їх ідентифікації, упорядкованого зберігання та швидкого віднайдення в електронних ресурсах, передбачатиме класифікування і пошук за різними критеріями (відповідно до специфіки електронних пошукових систем), дозволить вільно оперувати електронними документами незалежно від зміни їх типів, видів тощо.

Отже, створення інструментальних засобів для налаштування інформаційних реєстрів — актуальна задача на сьогоднішній день, адже неперервне збільшення кількості інформації в сучасному медіапросторі та швидкість цього збільшення потребують появи нових інструментів, що мають надавати змогу користувачам з будь-яким рівнем підготовки швидко та якісно отримувати необхідну інформацію, та збільшувати кількість якісно зібраних реєстрів за одиницю часу.

Метою даної роботи є програмна реалізація інструментальних засобів автоматичного налаштування динамічних реєстрів електронних інформаційних ресурсів.

Завдання полягає у створенні додатку для персональних комп’ютерів, який дозволяє користувачам швидко та без попередньої підготовки складати зручні каталоги масивів інформації, що призводить до значного зменшення часу виконання та опрацювання електронних ресурсів.

Об’єктом дослідження даної роботи є динамічні реєстри інформації. Предметом дослідження даної роботи є інструментальні засоби та системи керування електронними інформаційними ресурсами та динамічними реєстрами які вони складають.

В результаті роботи, набуло подальшого розвитку використання клієнт-серверного рішення з використанням технології REST для транспортного та архітектурного стиля. Розроблена система поєднує досвід розробки потужних фреймворків як для відображення інформації, так і для її обробки та збереження.

Магістерська дисертація включає п’ять розділів. Перший розділ описує постановку задачі та проблеми, які треба вирішити. У другому розділі описуються алгоритми, які використовуються в системі, їхні математичні моделі та методи їхньої паралельної реалізації. У третьому розділі проводиться огляд існуючих рішень, їхніх

переваг та недоліків, та порівняння з рішенням, що розроблюється. Четвертий розділ включає опис засобів розробки та основних технологій, що використовувались при розробці системи, наведено опис програмної реалізації: структуру програмного продукту та модулі програми, методику роботи користувача з програмною системою та графічний інтерфейс користувача. Останній розділ складає розробку стартап проекту.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

В наш час, інформація створюється і розповсюджується дуже швидко. Поява комп'ютерів, розвиток технологій комунікацій та збільшення технічного облаштування у кожної людини спричинили інформаційну сингулярність: дуже багато ресурсів знань з'являються і зникають водночас. Віднайти, відфільтрувати серед численних джерел і запам'ятати ресурс – становиться дедалі важче і важче. Проте ще більш складна задача – це підтримувати такі каталоги електронних інформаційних ресурсів та, відповідно, орієнтуватись в них в умовах постійного доповнення і оновлення.

Динамічна система — математична абстракція, в широкому розумінні довільна (зокрема, фізична, економічна) система, що змінюється в часі. Прикладом можуть бути механічні системи (рухомі групи тіл) або фізичні процеси.

Ознаками динамічності реєстрів можуть бути:

- Темпоральні ознаки – такі ознаки, які змінюються з часом, наприклад: актуальність інформаційних ресурсів у реєстрі (тобто актуальність самого реєстру), цінність реєстру, загальна кількість інформаційних ресурсів в реєстрі.

- Частотні ознаки – такі ознаки, які змінюються за певною частотою, наприклад: частота використання реєстру, кількість посилань на реєстр(частота цитування).

- Атрибутивні ознаки – такі ознаки, які залежать від кількості різнотипної мета-інформації в реєстрі: кількісно-динамічні атрибути (кількість постійно змінюється), якісно-динамічні атрибути (значення атрибутів постійно змінюється, наприклад з'являються нові інформаційні ресурси в реєстрі).

1.1. Особливості електронних інформаційних ресурсів у файлових системах

З появою комп'ютерів з'явилась задача збереження даних. Для її вирішення, було розроблено файлові системи, які могли б дати зручний інтерфейс користувачу для керуванням даними. Без файлових систем, інформація, що зберігається на носіях,

була б один великим шматком бітів, в яких було б важко розрізнити кінець одного ресурсу і початок іншого. Розділяючи інформацію на частини і надаючи кожній частині унікальний ідентифікатор, дані легко ізолювати та ідентифікувати. Історично склалось, що такі відокремлені частини інформації називають подібно до паперового аналогу – файлами.

Файлові системи можна використовувати на численних різних типах пристроїв зберігання даних, які використовують різні типи носіїв. Найпоширеніший пристрій зберігання даних, що використовується - це жорсткий диск. Інші види використовуваних носіїв включають флеш-пам'ять, магнітні стрічки та оптичні диски. У деяких випадках, як, наприклад, з tmpfs («temporary file system» - тимчасова файлова система у оперативній пам’яті), основна пам'ять комп'ютера (оперативна пам'ять, оперативна пам'ять) використовується для створення тимчасової файлової системи для короткочасного використання.

Деякі файлові системи використовуються на локальних пристроях зберігання даних [2], інші надають доступ до файлів через мережевий протокол (наприклад, NFS, [3] SMB або 9P клієнти). Деякі файлові системи є "віртуальними", а це означає, що надані "файли" (називаються віртуальними файлами) обчислюються за запитом (наприклад, procfs і sysfs) або просто відображаються в іншу файлову систему, що використовується як резервний магазин. Файлова система керує доступом як до вмісту файлів, так і до метаданих про ці файли. Вона несе відповідальність за організацію місця для зберігання.

Файлова система складається з двох або трьох шарів (рис. 1.1). Іноді шари явно відокремлюються, і іноді функції об'єднуються [4].

Логічна файлова система відповідає за взаємодію з користувальницькою програмою. Він забезпечує інтерфейс прикладної програми (API) для файлових операцій - OPEN, CLOSE, READ і т. Д., І передає запитувану операцію до шару під ним для обробки. Логічна файлова система "керує відкритими записами файлової таблиці та дескрипторами файлів для кожного процесу" [5]. Цей шар забезпечує "доступ до файлів, операції з каталогами, безпека та захист" [4].

Другий необов'язковий шар - це віртуальна файлова система. "Цей інтерфейс дозволяє підтримувати декілька паралельних екземплярів фізичних файлових систем, кожен з яких називається реалізацією файлової системи" [5].

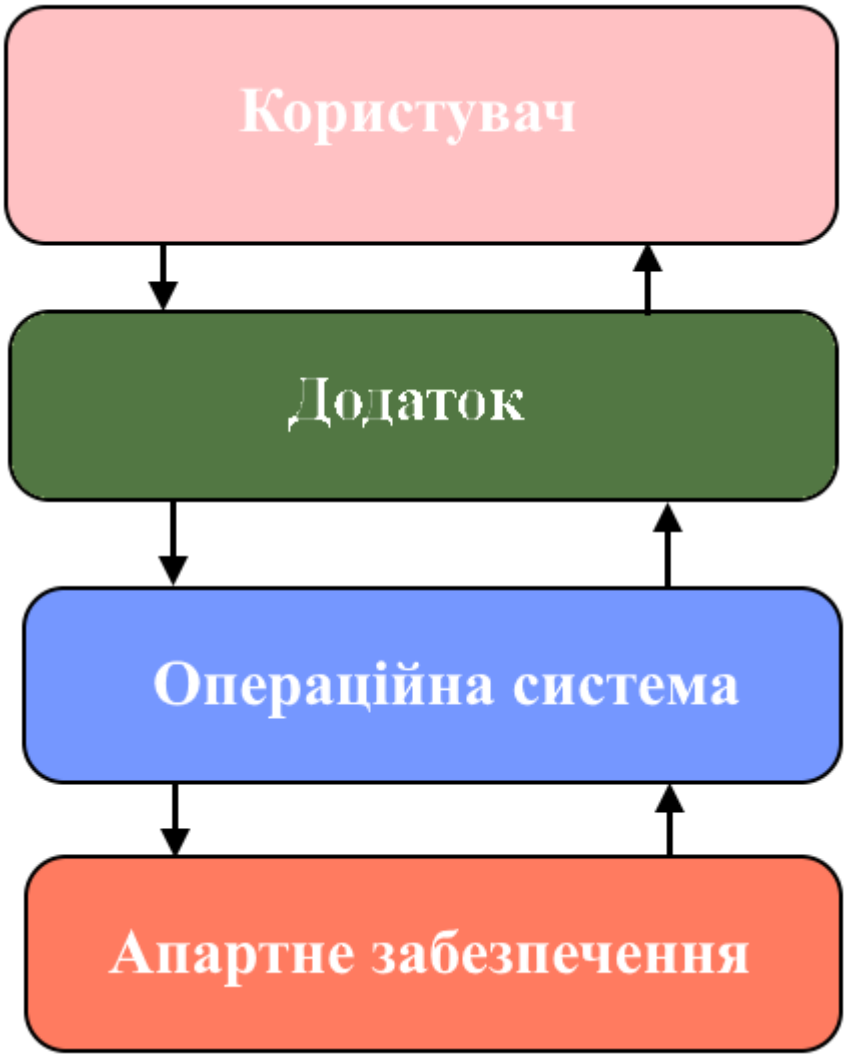

Рисунок 1.1 – Компоненти файлової системи

Третій шар - це фізична файлова система. Цей шар стосується фізичної роботи пристрою зберігання даних (наприклад, диска). Він обробляє фізичні блоки, які читаються чи записуються. Він обробляє буферизацію та керування пам'яттю і несе відповідальність за фізичне розміщення блоків у певних місцях на носії. Фізична файлова система взаємодіє з драйверами пристроїв або з каналом для керування пристроєм зберігання даних [4].

Файлові системи виділяють простір гранульованим способом, як правило, декількома фізичними одиницями на пристрої. Файлова система відповідає за

організацію файлів та каталогів, а також стеження за тим, які області медіа-файлів належать до якогось файлу і які не використовуються. Наприклад, в Apple DOS початку 1980-х років 256-байтові сектори на 140-кілобайтовій дискеті використовували сектора.

Це призводить до невикористаного простору, коли файл не точно кратний блоку розподілу, іноді він називається пропуском. Для розподілу 512 байт середнє значення невикористаного простору становить 256 байт. Для кластерів 64 Кб середня невикористана площа становить 32 КБ. Розмір блоку виділення вибирається при створенні файлової системи. Вибір розміру залежно від середнього розміру файлів, які очікуються в файловій системі, може мінімізувати кількість непридатних для використання простору. Часто розподіл за умовчанням може забезпечити розумне використання. Вибір розміру, який занадто малий, призводить до надмірних накладних витрат, якщо файлова система буде містити в основному дуже великі файли.

Фрагментація файлової системи виникає, коли невикористовуване місце або окремі файли не сусідні. Коли використовується файлова система, файли створюються, змінюються та видаляються. Коли файл створюється, файлова система виділяє простір для даних. Деякі файлові системи дозволяють або вимагають вказати початкове розподілення пробілів та наступні додаткові розподіли при збільшенні файлу. Оскільки файли видаляються, простір, який вони виділяють, остаточно розглядається як доступний для використання іншими файлами. Це створює чергування використаних і невикористовуваних областей різного розміру. Це фрагментація вільного простору. Коли файл створюється, і для його початкового розподілу не існує простору суміжного простору, простір слід призначити фрагментами. Коли файл модифікується таким чином, що він стає більшим, він може перевищувати спочатку призначене для нього місце, інший розподіл слід призначити деінде, і файл стає фрагментарним.

Назва файлу (або ім'я файлу) використовується для ідентифікації місця зберігання у файловій системі. Більшість файлових систем мають обмеження на тривалість імен файлів. У деяких файлових системах імена файлів не є чутливими до регістру (тобто

імена MYFILE і myfile відносяться до одного файлу); в інших імена файлів чутливі до регістру (наприклад, імена MYFILE, MyFile і myfile відносяться до трьох окремих файлів).

Файлові системи зазвичай мають каталоги (також називаються папками), які дозволяють користувачеві групувати файли в окремі колекції. Це може бути реалізовано шляхом асоціювання імені файлу з індексом у змісті або inode у файлової системи Unix. Структура каталогів може бути плоскою (тобто лінійною), або дозволяти ієрархії, де каталоги можуть містити підкаталоги. Перша файлова система для підтримки довільної ієрархії каталогів була використана в операційній системі Multics [6]. Національні файлові системи Unix-подібних систем також підтримують довільні ієрархії каталогів, як це відбувається, наприклад, з ієрархічною файловою системою Apple і її наступником HFS + у класичній Mac OS, файловою системою FAT в MS-DOS 2.0 та пізніших версіях MS- DOS і в Microsoft Windows, файлову систему NTFS в сімействі операційних систем Windows NT, ODS-2 (On-Disk Structure-2) і більш високі рівні файлової системи Files-11 у OpenVMS.

Інша бухгалтерська інформація зазвичай асоціюється з кожним файлом у файловій системі. Тривалість даних, що містяться в файлі, може зберігатися як кількість блоків, що виділяються для файлу або як кількість байтів. Час останнього змінення файлу може зберігатися як мітка часу файлу. Файлові системи можуть зберігати час створення файлу, час останнього доступу, час зміни метаданих файлу або час резервного копіювання файлу. Інша інформація може включати тип пристрою файлу (наприклад, блок, символ, сокет, підкаталог тощо), його ідентифікатор власника та ідентифікатор групи, його права доступу та інші атрибути файлів (наприклад, чи файл є тільки для читання, виконуваний файл тощо).

Файлова система зберігає всі метадані, пов'язані з файлом, включаючи ім'я файлу, довжину вмісту файлу та розташування файлу в ієрархії папки, окремо від вмісту файлу.

Більшість файлових систем зберігає імена всіх файлів в одному каталозі в одному місці - таблицю каталогів для цього каталогу, яка часто зберігається, як і будь-який інший файл. Багато файлові системи поставляють лише деякі метадані для файлу в

таблицю каталогів, а решту метаданих для цього файлу в цілком окремій структурі, такій як inode.

Більшість файлових систем також зберігають метадані, не пов'язані з жодним конкретним файлом. Такі метадані містять інформацію про невикористані простори растрового зображення, карту наявності блоків та інформацію про погані сектори. Часто така інформація про групу розподілу зберігається всередині самої групи розподілу.

Додаткові атрибути можуть бути пов'язані з файловими системами, такими як NTFS, XFS, ext2, ext3, деякими версіями UFS і HFS +, використовуючи розширені атрибути файлів. Деякі файлові системи передбачають призначені користувачем атрибути, такі як автор документа, кодування символу документа чи розмір зображення. Деякі файлові системи дозволяють зв'язати різні набори даних з одним іменем файлу.

Ці окремі колекції можна називати потоками або вилами. Компанія Apple вже давно використовує файлову систему на Macintosh, а Microsoft підтримує потоки в NTFS. Деякі файлові системи підтримують кілька попередніх змін файлу під одним ім'ям файлу; ім'я файлу самостійно отримує найновішу версію, тоді як попередню збережену версію можна отримати за допомогою спеціальної конфігурації назв, наприклад "filename; 4" або "filename (-4)", щоб отримати доступ до версії чотири заощадження тому.

Більшість сучасних файлових систем дозволяють іменам файлів містити широкий діапазон символів з набору символів Unicode. Однак вони можуть мати обмеження на використання певних спеціальних символів, забороняючи їх в іменах файлів; ці символи можуть бути використані для позначення пристрою, типу пристрою, префікса каталогу, роздільника шляху до файлу або типу файлу.

Файлові системи включають утиліти для ініціалізації, зміни параметрів і видалення екземпляра файлової системи. Деякі включають можливість поширювати або скоротити пробіл, що виділяється для файлової системи.

Каталог утиліт може бути використаний для створення, перейменування та видалення записів в каталозі, які також відомі як dentries [7], а також змінювати

метадані, пов'язані з каталогом. Утиліти каталогів також можуть включати можливості для створення додаткових посилань на каталог (жорсткі посилання в Unix), перейменування батьківських посилань («..» в Unix-подібних операційних системах), та створення двонаправлених посилань на файли.

Файлові утиліти створюють, перераховують, копіюють, переміщують та видаляють файли та змінюють метадані. Вони можуть скоротити дані, скоротити або розширити простір, додавати, переміщати та змінювати файли на місці. Залежно від базової структури файлової системи, вони можуть передбачати механізм для додавання або стиснення з початку файлу, вставки записів у середину файлу або видалення записів з файлу. Служби для вільного простору для видалених файлів, якщо файлова система надає функцію відновити, також належать до цієї категорії. Деякі файлові системи відкладають такі операції, як реорганізація вільного простору, безпечне видалення вільного простору та перебудова ієрархічних структур, надаючи утиліти для виконання цих функцій під час мінімальної активності. Прикладом є утиліти дефрагментації файлової системи.

Деякі найважливіші функції утиліт файлової системи включають в себе наглядові дії, які можуть включати в себе обхід права власності або прямий доступ до базового пристрою. До них відносяться високопродуктивне резервне копіювання та відновлення, реплікація даних та реорганізація різних структур даних і таблиць розподілу в файловій системі.

Однією з важливих функцій файлової системи є забезпечення того, що, незалежно від дій програм, що отримують доступ до даних, структура залишається постійною. Це включає в себе дії, вчинені, якщо дані, що змінюють програму, закінчуються ненормально, щоб інформувати файлову систему про те, що вона завершила свою діяльність. Це може включати оновлення метаданих, запис каталогу та обробку будь-яких даних, які були буферизовані, але ще не оновлені на фізичному носії інформації.

Інші проблеми, з якими повинна займатися файлова система, включають відмову від передачі даних або втрату з'єднання з віддаленими системами. У разі відмови операційної системи або "м'якої" відмови, виклик спеціальних процедур у файловій системі повинен бути подібним до того, коли програма не працює. Файлова система

також має змогу виправляти пошкоджені структури. Вони можуть виникнути внаслідок несправності операційної системи, для якої ОС не вдалося повідомити про файлову систему, про збої живлення або про скидання. Файлова система також повинна записувати події, щоб дозволити аналіз системних проблем, а також проблеми з конкретними файлами або каталогами.

Найважливішою метою файлової системи є керування користувацькими даними. Це включає в себе зберігання, вилучення та оновлення даних. Деякі файлові системи приймають дані для зберігання як потік байтів, які збираються та зберігаються таким чином, щоб вони були ефективними для носія. Коли програма отримує дані, вона визначає розмір буфера пам'яті, а файлова система передає дані з носія в буфер. Програма бібліотечної структури іноді може дозволити користувацькій програмі визначати запис на основі виклику бібліотеки, що визначає довжину. Коли програма читає дані, бібліотека отримує дані через файлову систему та повертає запис.

Деякі файлові системи дозволяють специфікувати фіксовану довжину запису, яка використовується для всіх записів і читання. Це полегшує розміщення n-й запису, а також оновлення записів. Ідентифікація для кожного запису, також відома як ключ, робить для більш складної файлової системи. Програма користувача може читати, записувати та оновлювати записи незалежно від їх розташування. Це вимагає складного управління блоками засобів масової інформації, які зазвичай розділяють ключові блоки та блоки даних. Дуже ефективні алгоритми можуть бути розроблені з пірамідною структурою для пошуку записів. [8]

Часто системи роздрібної торгівлі налаштовуються з єдиною файловою системою, що займає весь накопичувач.

Інший підхід полягає в тому, щоб розділити диск так, щоб можна було використовувати декілька файлових систем з різними атрибутами. Одна файлова система, яка використовується як кеш-пам'ять браузера, може бути налаштована з невеликим розміром розміру. Це має додаткову перевагу: зберігати шаленою активність створення та видалення файлів, типових для роботи браузера, у вузькій області диску, і не заважати розподілу інших файлів. Подібний розділ може бути створений для електронної пошти. Інший розділ та файлова система можуть бути

створені для зберігання аудіо- або відеофайлів з відносно великим розподілом. Один з файлових систем, як правило, може бути встановлений лише для читання і лише періодично може бути встановлений для запису.

Третій підхід, який в основному використовується в хмарних системах, полягає в тому, щоб використовувати "образ диска" для розміщення додаткових файлових систем із однаковими атрибутами чи ні в іншій (хостовій) файловій системі як файл. Загальним прикладом є віртуалізація: один користувач може запустити експериментальний дистрибутив Linux (використовуючи файлову систему ext4) на віртуальній машині під його виробничим середовищем Windows (за допомогою NTFS). Файлова система ext4 розміщується на зображенні диску, який розглядається як файл (або кілька файлів, залежно від гіпервізора та параметрів) у файловій системі хоста NTFS.

Використання декількох файлових систем в одній системі має додаткову перевагу, що в разі пошкодження одного розділу, інші файлові системи часто залишаються незмінними. Це включає в себе знищення вірусу системного розділу або навіть систему, яка не завантажиться. Утиліти файлової системи, які вимагають спеціального доступу, можуть бути ефективно виконані поетапно. Крім того, дефрагментація може бути більш ефективною. Деякі утиліти обслуговування системи, такі як сканування та резервне копіювання вірусів, також можуть бути оброблені в сегментах. Наприклад, не потрібно резервувати файлову систему, що містить відео разом з усіма іншими файлами, якщо ніхто не був доданий після останньої резервної копії. Що стосується файлів зображень, то можна легко "вимкнути" диференціальні зображення, які містять лише "нові" дані, записані на основне (оригінальне) зображення. Диференціальні зображення можуть використовуватися для обох проблем безпеки (як "одноразові" системи - їх можна швидко відновити, якщо вони знищені або забруднені вірусом, оскільки старе зображення можна видалити, а новий образ можна створити за декілька секунд, навіть без автоматизовані процедури) та швидке розгортання віртуальної машини (оскільки різноманітні зображення можуть швидко створюватися за допомогою сценарію у партіях).

Отже, можна побачити, що файлові системи доволі складні та різнотипні. А це означає, що для даної роботи необхідно мати універсальний метод доступу до файлу – незалежно від способу організації файлової системи. Проте, не все електронні інформаційні ресурси знаходяться у локальному доступі, тому звернемось до мережевих типів ресурсів.

1.2. Особливості електронних інформаційних ресурсів у комп'ютерній мережі

В сучасних способах організації мережевих систем, користувачі мають невеликий набір методів доступу ти видів відображення віддалених електронних інформаційних ресурсів. Найбільш поширеними є веб-сайти.

Веб-сайт [9] - це сукупність відповідних веб-сторінок, включаючи мультимедійний вміст, типово ідентифікований з загальним доменним іменем, і публікується щонайменше на одному веб-сервері. Примітні приклади: wikipedia.org, google.com і amazon.com. Сьогодні приблизно 380 нових веб-сайтів створюються щотижня в усьому світі. [10]

Веб-сайт може бути доступним через загальнодоступну мережу Інтернет-протоколу (IP), таку як Інтернет, або приватну локальну мережу (LAN), посилаючись на уніфікований локатор ресурсів (URL-адресу), який ідентифікує сайт.

Веб-сайти можуть мати багато функцій і можуть бути використані в різних модах; веб-сайт може бути персональним веб-сайтом, корпоративним веб-сайтом компанії, державним веб-сайтом, веб-сайтом організації тощо. Веб-сайти, як правило, присвячені певній темі або цілям, починаючи від розважальних та соціальних мереж, а також надаючи новини та освіту. Всі загальнодоступні веб-сайти колективно складають Всесвітню мережу, а приватні веб-сайти, такі як веб-сайт компанії для своїх співробітників, як правило, є частиною інтрамережі.

Веб-сторінки, які є будівельними блоками веб-сайтів, - це документи, які зазвичай складаються з простого тексту, що містяться в інструкціях форматування гіпертекстової розмітки (HTML, XHTML). Вони можуть включати елементи з інших

веб-сайтів із відповідними якорями розмітки. Веб-сторінки доступні та транспортуються за допомогою протоколу передачі гіпертексту (HTTP), який за бажанням може використовувати шифрування (HTTP Secure, HTTPS) для забезпечення безпеки та конфіденційності для користувача. Застосування користувача, часто веб-браузер, перетворює вміст сторінки відповідно до інструкцій розмітки HTML на дисплейний термінал.

Гіперпосилання між веб-сторінками передає читачеві структуру сайту та спрямовує навігацію сайту, що часто починається з домашньої сторінки, що містить каталог веб-контенту сайту. Деякі веб-сайти вимагають реєстрації користувача або підписки на доступ до вмісту. Приклади веб-сайтів підписки включають безліч бізнес-сайтів, веб-сайти новин, веб-сайти академічних журналів, ігрові веб-сайти, веб-сайти для обміну файлами, дошки оголошень, електронна пошта на веб-сайтах, веб-сайти соціальних мереж, веб-сайти, що надають дані в реальному часі на фондовому ринку, а також сайти, що надають різні інші послуги. Кінцеві користувачі можуть отримувати доступ до веб-сайтів на різних пристроях, включаючи настільні та портативні комп'ютери, планшетні комп'ютери, смартфони та інтелектуальні телевізори.

Веб-сайти мають багато функцій і можуть бути використані в різних режимах; веб-сайт може бути персональним веб-сайтом, комерційним веб-сайтом, державним веб-сайтом або веб-сайтом некомерційної організації. Веб-сайти можуть бути роботами фізичної особи, бізнесу або іншої організації, і, як правило, присвячені певній темі або цілям. Будь-який веб-сайт може містити гіперпосилання на будь-який інший веб-сайт, тому різниця між окремими сайтами, сприйманими користувачем, може бути розмитою. Веб-сайти записуються або конвертуються в HTML (Мова розмітки гіпертексту), і доступ до них здійснюється за допомогою інтерфейсу програмного забезпечення, класифікованого як користувацький агент. Веб-сторінки можна переглядати або отримувати в інший спосіб із різних комп'ютерів та інтернет-пристроїв різного розміру, включаючи настільні комп'ютери, ноутбуки, планшетні комп'ютери та смартфони. Веб-сайт розміщується на комп'ютерній системі, відомій як веб-сервер, також називається HTTP-сервер (протокол передачі гіпертексту). Ці

терміни також можуть посилатися на програмне забезпечення, яке працює на цих системах, яке витягує та передає веб-сторінки у відповідь на запити користувачів веб-сайту. Apache є найбільш часто використовуваним веб-серверним програмним забезпеченням (за статистикою Netcraft), IIS Microsoft також широко використовується. Деякі альтернативи, такі як Nginx, Lighttpd, Hiawatha або Cherokee, повністю функціональні та легкі.

Веб-сайти можна розділити на дві широкі категорії - статичні та інтерактивні. Інтерактивні сайти є частиною спільноти веб-сайтів 2.0 і дозволяють інтерактивність між власником сайту та відвідувачами сайту або користувачами. Статичні сайти служать або фіксують інформацію, але не дозволяють взаємодіяти безпосередньо з аудиторією чи користувачами. Деякі веб-сайти є інформаційними або створені ентузіастами або для особистого користування чи розваг. Багато веб-сайтів прагнуть заробляти гроші, використовуючи одну або декілька бізнес-моделей, зокрема:

- публікація цікавого контенту та продаж контекстної реклами через прямі продажі або через рекламну мережу;
- електронна комерція: продукти чи послуги купуються безпосередньо через веб-сайт;
- рекламні товари або послуги, доступні в цегельному та мінометному бізнесі;
- Freemium: основний вміст доступний безкоштовно, але для преміум-вмісту потрібен платіж (наприклад, веб-сайт WordPress, це платформа з відкритим вихідним кодом для створення блогу чи веб-сайту).

Є багато різновидів веб-сайтів, кожен із яких спеціалізується на певному типі вмісту чи використання, і вони можуть бути довільно класифіковані в будь-якій кількості способів.

Деякі веб-сайти можуть бути включені в одну або декілька інформаційних категорій. Наприклад, веб-сайт для бізнесу може рекламувати продукти компанії, але також може містити інформаційні документи, такі як white paper. Існує також численні підкатегорії, перераховані вище. Наприклад, порносайт є певним типом веб-сайту електронної комерції або бізнес-сайту (тобто він намагається продати членство

для доступу до свого сайту) або мають можливості соціальних мереж. Фанатик може бути відданістю від власника до певної знаменитості.

Веб-сайти обмежені архітектурними обмеженнями (наприклад, обчислювальні потужності, присвячені веб-сайту). Дуже великі веб-сайти, такі як Facebook, Yahoo!, Microsoft та Google, використовують безліч серверів і обладнання для балансування навантаження, такі як Cisco Content Services Switches, для розповсюдження завантажень відвідувачів на декількох комп'ютерах у різних місцях. На початку 2011 року в Facebook було використано 9 центрів обробки даних з приблизно 63000 серверами.

У лютому 2009 року Netcraft, компанія з моніторингу Інтернету, яка відстежила зростання веб-сайтів з 1995 року, повідомила, що в 2009 році на них було 215 675 903 веб-сайти з доменними іменами та вмістом, у порівнянні з 19 732 веб-сайтами в серпні 1995 року. [11] Після досягнення 1 мільярда веб-сайтів у вересні 2014 року, підтверджений компанією NetCraft в огляді веб-серверів у жовтні 2014 року, і що Інтернет-статистики в мережі Інтернет було першим, хто оголосив про це, як це засвідчили цей твіт, від винахідника самого Всесвітньої павутини Тим Бернерс- Лі - кількість веб-сайтів в усьому світі згодом зменшилась, повернувшись до рівня нижче 1 мільярда. Це пов'язано з щомісячними коливаннями кількості неактивних веб-сайтів. Кінець веб-сайтів до березня 2016 року продовжував збільшуватися до 1 мільярда, і з тих пір продовжує зростати. [12]

Ранні сайти мали лише текст, а незабаром - зображення. Плагіни веб-браузера були потім використані для додавання аудіо, відео та інтерактивності (наприклад, для багатих Інтернет-додатків, що відображає складність настільної програми, наприклад, текстовий процесор). Прикладами таких плагінів є Microsoft Silverlight, Adobe Flash, Adobe Shockwave та аплети, написані на Java. HTML 5 містить положення для аудіо та відео без плагінів. JavaScript також вбудований у більшість сучасних веб-переглядачів, і дозволяє творцям веб-сайтів надсилати коди в веб-браузері, в якому вказується, як інтерактивно змінювати вміст сторінки та спілкуватися з веб-сервером, якщо це необхідно. Внутрішнє представлення вмісту веб-переглядача відома як Model Object Model (DOM), і ця техніка називається

Dynamic HTML. Тенденція у 2010 році на веб-сайтах під назвою "чуйний дизайн" дала найкращий досвід перегляду, оскільки вона забезпечує макет для пристроїв на базі користувачів. Ці веб-сайти змінюють макет відповідно до пристрою або мобільної платформи, що дає багатий досвід користувача. [13]

Динамічний веб-сайт, який часто і автоматично змінює або налаштовує себе. Динамічні сторінки на стороні сервера створюються "на льоту" за допомогою комп'ютерного коду, який створює HTML (CSS несуть відповідальність за зовнішній вигляд і, таким чином, є статичними файлами). Існує широкий спектр програмних систем, таких як CGI, Java Servlet і Java Server Pages (JSP), Active Server Pages та ColdFusion (CFML), доступні для створення динамічних веб-систем та динамічних сайтів. Різні веб-додатки та системи веб-шаблонів доступні для загальнодоступних мов програмування, таких як Perl, PHP, Python та Ruby, щоб швидше і простіше створювати складні динамічні веб-сайти.

Сайт може відображати поточний стан діалогу між користувачами, відслідковувати ситуацію, що змінюється, або надавати певну інформацію індивідуально для потреб конкретного користувача. Наприклад, коли запитується перша сторінка сайту новин, код, що запускається на веб-сервері, може об'єднати збережені фрагменти HTML із новинами, завантаженими з бази даних або іншого веб-сайту через RSS, щоб створити сторінку, яка містить найновішу інформацію. Динамічні сайти можуть бути інтерактивними, використовуючи HTML-форми, зберігаючи та читаючи файли cookie-файлів, або створюючи серію сторінок, що відображають попередню історію кліків.

Іншим прикладом динамічного контенту є те, коли роздрібний веб-сайт із базою даних засобів масової інформації дозволяє користувачеві вводити запит на пошук, наприклад. за ключовим словом Beatles. У відповідь вміст веб-сторінки спонтанно зміниться так, як він виглядав раніше, а потім відображатиме список продуктів Beatles, таких як компакт-диски, DVD-диски та книги. Динамічний HTML використовує код JavaScript, щоб вказувати веб-браузер, як інтерактивно змінювати вміст сторінки. Один із способів симулювати певний тип динамічного веб-сайту, уникаючи при цьому втрати продуктивності від ініціювання динамічного двигуна на

основі кожного користувача або підключення, - це періодично автоматично відтворювати велику серію статичних сторінок.

Відповідно, доступ до електронних інформаційних ресурсів можна отримати за допомогою файлових систем. Проте не завжди файли знаходяться у локальному доступі, тому користувач повинен використовувати веб-переглядач для доступу до інформаційних ресурсів, що знаходяться у мережі. Єдине, що об’єднує спосіб доступу до файлів у файловій системі і доступ до мережевих ресурсів – посилання. Тому було вирішено створити у цій роботі певне сховище таких посилань, яке було б автоматизованим та піддавалось користувацькому налаштуванню.

У даній роботі розглядаються деякі найрозповсюдженіші методи класифікації, а також досліджуються найефективніші засоби реалізації алгоритмів їх застосування. Кожний електронний ресурс має деякі загальні та свої унікальні метадані, які можна проаналізувати і віднести його до певного класу.

Помітно, що попри нескладні алгоритми обробки кожного електронного ресурсу, налаштування цілого реєстру що постійно змінюється у реальному часі є відносно повільним, адже одне реєстр може складатися з десятків мільйонів записів, а якщо робити пакетну обробку, то питання часу постає досить гостро. Таким чином, необхідно розробити максимально ефективний алгоритм класифікації. Цього можна досягти за допомогою правильного налаштування динамічних реєстрів, що зробить роботу з ними набагато ефективнішою і зручнішою.

2. АЛГОРИТМИ КЛАСИФІКАЦІЇ І КЛАСТЕРИЗАЦІЇ

Існує значна кількість алгоритмів, які працюють з інформаційними ресурсами. В даній роботі розглянуто основні методи класифікації, кластеризації та способи групування інформації у кластери.

2.1. Основні підходи до класифікації інформаційних ресурсів

Існують три підходи до задачі класифікації [14].

По-перше, класифікація не завжди здійснюється за допомогою комп'ютера. Наприклад, у звичайній бібліотеці тематичні рубрики присвоюються книгам власноруч бібліотекарем. Подібна ручна класифікація дорога і непридатна у випадках, коли необхідно класифікувати велику кількість документів з високою швидкістю.

Інший підхід полягає в написанні правил, згідно яких можна зарахувати текст до тієї чи іншої категорії. Наприклад, одне з таких правил може виглядати наступним чином: «якщо текст містить слова похідна і рівняння, то віднести його до категорії математика». Спеціаліст, який знайомий з предметною областю і володіє навичкою написання регулярних виразів, може скласти низку правил, які потім автоматично застосовуються до класифікації нових документів. Цей підхід кращий, ніж попередній, оскільки процес класифікації автоматизується і кількість оброблюваних документів стає практично не обмеженою. Більш того, побудова правил власноруч може підвищити точність класифікації у порівнянні з машинним навчанням (див. нижче). Однак створення і підтримка правил в актуальному стані (наприклад, якщо для класифікації новин використовується ім'я чинного президента країни, то відповідне правило потрібно час від часу змінювати) вимагає постійних зусиль фахівця.

Нарешті, третій підхід ґрунтується на машинному навчанні. У цьому підході набір правил або, більш загально, критерій прийняття рішення текстового класифікатора

обчислюється автоматично з навчальних даних (іншими словами, проводиться навчання класифікатора).

Навчальні дані — це деяка кількість наочних зразків документів з кожного класу. У машинному навчанні зберігається необхідність ручної розмітки (термін «розмітка» означає процес надання документу певного класу), але вона є більш простим завданням, ніж написання правил. Крім того, розмітка може бути проведена в звичайному режимі використання системи. Наприклад, у програмі електронної пошти може існувати можливість позначати листи як спам, таким чином формуючи навчальну множину для класифікатора — фільтра небажаних повідомлень. Тому класифікація текстів, заснована на машинному навчанні, є прикладом навчання з учителем, де в ролі вчителя виступає людина, що задає набір класів і розмічає навчальну множину.

Останніми роками значні зусилля були спрямовані на підвищення ефективності існуючих алгоритмів. [15] [16] Серед них є CLARANS (Ng і Han, 1994), [17] і BIRCH (Zhang et al., 1996). [18] З недавньою потребою обробляти більші та більші набори даних (також відомі як великі дані), зростає готовність торгувати семантичним значенням сформованих кластерів для продуктивності. Це призвело до розробки методів перед кластеризації, таких як кластерування куполів, які можуть ефективно обробляти величезні набори даних, але результуючі "кластери" - це лише грубе попереднє розділення набору даних для аналізу розділів з існуючими більш повільними методами як k-засобу кластеризації.

Для великогабаритних даних багато хто з існуючих методів не вдається через проблеми розмірності, що робить певні дистанційні функції проблематичними у високорозмірних просторах. Це призвело до нових алгоритмів кластеризації для великогабаритних даних, що зосереджуються на кластеризації підпрофільників (де використовуються лише деякі атрибути, а кластерні моделі включають відповідні атрибути для кластера) і кореляційне кластеризація, що також шукає довільне обернене ("корельоване") підпростір кластери, які можна моделювати шляхом співставлення їх атрибутів [19]. Прикладами таких кластерних алгоритмів є CLIQUE [20] і SUBCLU [21].

Ідеї з методів кластеризації на базі щільності (зокрема, сімейство алгоритмів DBSCAN / OPTICS) були прийняті до кластеризації підпрофілю (ієрархічний підпростір кластера HiSC, [22] і кореляційного кластування (HiCO [23]) ієрархічної кореляції кластеризації, 4C [24] за допомогою "кореляційного підключення" та ERiC [25], що досліджує ієрархічні кореляційні кластери на основі щільності).

Було запропоновано декілька різних кластерних систем на основі взаємної інформації. Однією з варіантів інформаційної метрики є Маріна Мєлга, [26] інша забезпечує ієрархічну кластеризацію [27]. Використовуючи генетичні алгоритми, можна оптимізувати широкий діапазон різних придатних функцій, включаючи взаємну інформацію [28]. Крім того, алгоритми передачі повідомлень, останні розробки в галузі інформатики та статистичної фізики, призвели до створення нових типів кластерних алгоритмів [29].

Оцінка (або "перевірка") результатів кластеризації є такою ж складною, як і кластеризація. [30] Популярні підходи включають "внутрішню" оцінку, де кластеризація підсумовується до єдиної оцінки якості, "зовнішньої" оцінки, де класифікація порівнюється з існуючою класифікацією "істинності землі", оцінкою "ручної" експертизи людини та "непрямим "оцінку шляхом оцінки корисності кластеризації в його передбачуваному застосуванні [31].

Внутрішні заходи оцінювання страждають від проблеми, що вони представляють функції, які самі можуть розглядатися як кластеризована мета. Наприклад, можна згрупувати дані, задані коефіцієнтом Силуэта; крім того, що для цього не існує відомий ефективний алгоритм. Використовуючи такий внутрішній засіб для оцінки, досить порівнює схожість задач оптимізації [31] і не обов'язково, наскільки корисним є кластеризація.

Отже, жоден із цих підходів не може судити про фактичну якість кластеризації, але це потребує оцінки людей [31], що є дуже суб'єктивним. Тим не менш, така статистика може бути досить інформативною для виявлення поганих кластерингів [32], але не слід звільняти суб'єктивну оцінку людини [32].

Коли кластерний результат оцінюється на основі кластеризованих даних, це називається внутрішньою оцінкою. Ці методи зазвичай накладають найкращий

результат на алгоритм, який створює кластери з високою подібністю всередині кластера та низькою подібністю між кластерами. Один недолік застосування внутрішніх критеріїв оцінки кластерів полягає в тому, що високі показники за внутрішніми показниками не обов'язково призводять до ефективного використання інформаційних пошуків [33]. Крім того, ця оцінка є упередженим по відношенню до алгоритмів, які використовують ту ж кластерну модель. Наприклад, кластери k-tools природно оптимізують відстані об'єкта, а внутрішній критерій відстані, напевно, перевищує результуючу кластеризацію.

Тому заходи внутрішньої оцінки найкраще підходять для того, щоб отримати деяке уявлення про ситуації, коли один алгоритм виконує краще, ніж інший, але це не означає, що один алгоритм дає більш достовірні результати, ніж інший. Дійсність, яка вимірюється таким індексом, залежить від твердження, що така структура існує в наборі даних. Алгоритм, розроблений для деяких моделей, не має шансів, якщо набір даних містить принципово інший набір моделей, або якщо оцінка оцінює принципово інший критерій. Наприклад, у k-середніх кластерів можна знайти лише опуклий кластер, і багато показників оцінки вважають опуклими кластерами. На наборі даних з невипуклими кластерами ні використання к-засобів, ні критерію оцінки, що передбачає випуклості, не звучить.

При зовнішній оцінці результати кластеризації оцінюються на основі даних, які не використовувались для кластеризації, наприклад, відомих міток класу та зовнішніх орієнтирів. Такі орієнтири складаються з безлічі попередньо класифікованих предметів, і ці набори часто створюються (експертними) людьми. Таким чином, набори тестів можна розглядати як золотий стандарт для оцінки. Ці типи оціночних методів вимірюють, наскільки кластеризація близька до заздалегідь визначених еталонних класів. Проте останнім часом було обговорено, чи це адекватне для реальних даних, або тільки на синтетичних наборах даних з фактичною основою правди, оскільки класи можуть містити внутрішню структуру, наявні атрибути можуть не дозволяти розділення кластерів або класи можуть містити аномалії. Крім того, з точки зору відкриття знань, відтворення відомих знань не обов'язково може бути передбачуваним результатом. У спеціальному сценарії обмеженого

кластеризації, де метаінформація (наприклад, етикетки класів) використовується вже в процесі групування, тримання інформації для цілей оцінки є нетривіальним.

Ряд заходів адаптується з варіантів, що використовуються для оцінки завдань класифікації. Замість того, щоб підрахувати кількість разів, коли клас був правильно присвоєний єдиній точці даних (відомі як істинні позитиви), такі показники парних підрахунків визначають, чи передбачається, що кожна пара точок даних, дійсно в одному кластері, буде однаковою кластер.

2.2. Задача кластеризації та кластерний аналіз

Кластерний аналіз або кластеризація полягає в тому, щоб об'єднувати набір об'єктів таким чином, щоб об'єкти в тій самій групі (називається кластер) більш схожими (в певному сенсі) до один одного, ніж до інших груп (кластерів). Це головне завдання пошукового виявлення даних і загальна методика аналізу статистичних даних, що використовується у багатьох областях, включаючи машинне навчання, розпізнавання образів, аналіз зображень, пошук інформації, біоінформатика, стиснення даних та комп'ютерна графіка.

Сам кластерний аналіз - це не один конкретний алгоритм, а загальне завдання, яке потрібно вирішити. Це може бути досягнуто за допомогою різних алгоритмів, які значно відрізняються у розумінні того, що являє собою кластер, і як ефективно їх знайти. Популярні поняття кластерів включають групи з невеликими відстанями між членами кластера, щільними ділянками простору даних, інтервалами або певними статистичними розподілами. Кластеризація може бути сформульована як багатоцільова проблема оптимізації. Відповідний алгоритм групування та налаштування параметрів (включаючи такі параметри, як функція відстані для використання, порогове значення щільності чи кількість очікуваних кластерів) залежать від індивідуального набору даних та призначеного використання результатів. Кластерний аналіз як такий не є автоматичним завданням, а ітераційним процесом відкриття знань або інтерактивної багатоцільової оптимізації, що включає

пробну та неефективну роботу. Часто необхідно змінювати попередню обробку даних та параметри моделі, доки результат не досягне бажаних властивостей.

Крім кластеризації термінів існує ряд умов з аналогічними значеннями, включаючи автоматичну класифікацію, числову таксономію, ботріологію (від грецького βότρυς "виноград") та типологічний аналіз. Тонкі розбіжності часто полягають у використанні результатів: під час пошуку даних, отримані групи є предметом інтересу, в автоматичній класифікації це викликає інтерес до дискримінаційної сили.

Кластерний аналіз був започаткований в антропології Водієм і Крейбером у 1932 р. І ввів у психологію Зубіна в 1938 р. І Роберта Теріона в 1939 р. і знаменито використовується Каттелем, починаючи з 1943 р. для класифікації теорії ознак в психології особистості.

Поняття "кластеру" не можна точно визначити, що є однією з причин, чому існує так багато алгоритмів кластеризації. Існує загальний знаменник: група об'єктів даних. Проте різні дослідники використовують різні кластерні моделі, і для кожної з цих кластерних моделей можуть бути надані різні алгоритми. Поняття кластера, як встановлено різними алгоритмами, значно відрізняється від його властивостей. Розуміння цих "кластерних моделей" є ключовими для розуміння різниць між різними алгоритмами. Типові моделі кластерів включають:

- моделі з'єднання: наприклад, ієрархічна кластери створюють моделі на основі відстані зв'язку;

- моделі центроїдів: наприклад, алгоритм k-means представляє кожен кластер за допомогою одного середнього вектора;

- моделі розподілу: кластери моделюються за допомогою статистичних розподілів, таких як багатоваріантні нормальні розподіли, що використовуються алгоритмом максимізації очікування;

- моделі щільності: наприклад, DBSCAN та OPTICS визначають кластери як сполучені щільні області в просторі даних;

- моделі: підпростору в biclustering (також відома як спільне кластеризацию або Двухмодовой-кластеризації), кластери моделюються з обома членами кластера і відповідними атрибутами;

- моделі групи: деякі алгоритми не надають вишуканої моделі для своїх результатів і просто надають інформацію про групування;

- моделі Графіка на основі: кліку, тобто, підмножина вузлів в графі таким чином, що кожні два вузла в підмножині з'єднані ребром можна розглядати як прототип форму кластера.;

- нейронні моделі: найвідоміша неконтрольована нейронна мережа є кластером що самоорганізується і ці моделі зазвичай можуть бути охарактеризовані як схоже на одну або більше із зазначених вище моделей, і в тому числі моделей міжпросторових, коли нейронні мережі реалізовувати форму аналізу головних компонент або незалежний аналіз компонентів.

Кластеризація - сукупність таких кластерів, що зазвичай містять всі об'єкти в наборі даних. Крім того, він може визначати взаємозв'язок кластерів один з одним, наприклад, ієрархія кластерів, вбудованих один в одного. Кластеризації можна грубо розрізнити як:

- жорстка кластеризація: кожен об'єкт належить до кластера чи ні;

- м'які кластеризації (також: нечітка кластеризація): кожен об'єкт відноситься до кожного кластеру в певній мірі (наприклад, ймовірність того, що приналежність до групи).

Є також більш точні відмінності, наприклад:

- чіткий розподіл кластеризації: кожен об'єкт належить рівно одному кластеру;

- суворий розподіл кластеризації з викидами: об'єкти також можуть належати до жодного кластера, і вважаються витісненими;

- перекриття кластеризації (також: альтернативна кластеризація, кластеризація з декількома видами): об'єкти можуть належати до декількох кластерів; зазвичай за участю складних кластерів;

- ієрархічна кластеризація: об'єкти, що належать до дочірнього кластера, також належать батьківському кластеру;

- кластеризація підпрограм: при одночасно визначеному підпросторі кластерів, що перекриваються, кластери не повинні співпадати.

Як зазначено вище, алгоритми кластеризації можна класифікувати на основі їх кластерної моделі. У далі буде наведено лише найпомітніші приклади алгоритмів групування, оскільки існує, можливо, понад 100 опублікованих алгоритмів кластеризації. Не всі надають моделі для своїх кластерів і тому не можуть бути легко поділені на категорії.

Не існує об'єктивно "правильного" алгоритму кластеризації, але, як було зазначено, "кластеризація є в очах спостерігача". [4] Найбільш відповідний алгоритм кластеризації для конкретної задачі часто повинен бути обраний експериментально, якщо не існує математична причина, щоб віддавати перевагу однієї кластерної моделі над іншою. Слід зазначити, що алгоритм, розроблений для одного виду моделі, як правило, збігатиметься на наборі даних, що містить принципово інший вид моделі [4]. Наприклад, k-tools не може знайти непуклі кластери. [4]

Кластери на основі підключення, також відомі як ієрархічні кластери, базуються на основній ідеї об'єктів, більше пов'язаних з сусідніми об'єктами, а не до об'єктів далі. Ці алгоритми з'єднують "об'єкти" з утворенням "кластерів" на основі їх відстані. Кластер можна описати в основному як максимальну відстань, необхідну для підключення частин кластера. На різних відстанях утворюються різні кластери, які можуть бути представлені за допомогою дендрограми, що пояснює, звідки походить загальне ім'я "ієрархічне кластеризація": ці алгоритми не забезпечують жодного розбиття набору даних, а забезпечують велику ієрархію кластерів, що зливаються один з одним на певних відстанях. У дендрограмі вісь y позначає відстань, на якій кластери зливаються, тоді як об'єкти розміщуються уздовж вісі x таким чином, щоб кластери не змішувались.

Кластеризація на основі підключення - це ціла сімейство методів, які відрізняються тим, як обчислюються відстані. Окрім звичайного вибору віддалених функцій, користувач також повинен визначити критерій зв'язування (оскільки кластер складається з декількох об'єктів, є декілька кандидатів для обчислення відстані), що використовуються. Існують широко відомі варіанти, такі як кластеризація однієї зв'язки (мінімальна відстань об'єкта), повна кластеризація зв'язків (максимальна відстань об'єкта), а також UPGMA або WPGMA ("Невведений або зважений парний

метод з арифметичним середнім значенням", також відомий як середній зв'язок кластеризації). Крім того, ієрархічне кластування може бути агломераційним (починаючи з окремих елементів і об'єднуючи їх у кластери) або розподілу (починаючи з повного набору даних і поділяючи його на розділи).

Для того, щоб вирішити, які кластери повинні бути об'єднані (для агломеративних), або де кластер повинен бути розділений (для розбіжностей), потрібна міра несхожості між наборами спостережень. У більшості методів ієрархічної кластери це досягається використанням відповідної метрики (міри відстані між парами спостережень) і критерієм зв'язування, який вказує на несхожість множин як функцію попарних відстаней спостережень у множинах.

Вибір відповідної метрики вплине на форму кластерів, оскільки деякі елементи можуть бути близькими один до одного відповідно до однієї відстані та далі, відповідно до іншої. Наприклад, в 2-мірному просторі відстань між точкою (1,0) і початком (0,0) завжди дорівнює звичайним нормам, однак відстань між точками (1,1) і початком (0,0) може бути 2 у Манхеттенській відстані, 1.42 за евклідової відстані або 1 у максимальній відстані.

Деякі загальнодоступні показники для ієрархічного кластеризації наведені у таблиці (табл. 2.1). [4]

Для текстових або інших нечислових даних часто використовуються показники, такі як відстань Хеммінга або відстань Левенштейна.

Огляд кластерного аналізу в дослідженні психології здоров'я показало, що найпоширенішою мірою відстані в опублікованих дослідженнях у досліджуваній області є евклідова відстань або квадратична евклідова відстань.

Ієрархічна кластери мають чіткі переваги, що можуть бути використані будь-яка дієва мера відстані. Насправді самі спостереження не потрібні: все, що використовується, є матрицею відстаней.

Таблиця 2.1. – показники ієрархічної кластеризації

Назва	Формула
Ервклідова відстань	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Квадратична еквлідова відстань	$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
Манхеттенська відстань	$\|a-b\|_1 = \sum_i \|a_i - b_i\|$
Максимальна відстань	$\|a-b\|_\infty = \max_i \|a_i - b_i\|$
Відстань Махаланобіса	$\sqrt{(a-b)^\top S^{-1}(a-b)}$, де S - матриця коваріації

Де a – початкова точка, b – кінцева точка, i – відповідна проекція координати на вісь.

Окрім терміну кластеризація існує багато термінів з аналогічним значенням, серед яких автоматична класифікація, числова таксономія та типологічний аналіз. Тонкі розбіжності часто полягають у використанні результатів: для добування даних, отримані групи є предметом інтересу, при автоматичній класифікації, навпаки, більш важливий степінь розбіжності.

Основна мета кластерного аналізу — знаходження груп схожих об'єктів у вибірці. Спектр застосувань кластерного аналізу дуже широкий: його використовують в археології, антропології, медицині, психології, хімії, біології, державному управлінні, філології, маркетингу, соціології та інших дисциплінах. Однак універсальність застосування привела до появи великої кількості несумісних термінів, методів і підходів, що ускладнюють однозначне використання і несуперечливу інтерпретацію кластерного аналізу.

2.3 Нечітка кластеризація

Нечіткі кластери (також називаються м'яким кластеризацією або м'якими k-засобами) є формою кластеризації, в якій кожна точка даних може належати до декількох кластерів.

Кластеризація чи кластерний аналіз передбачає присвоєння точкам даних кластерів таким чином, щоб елементи одного кластеру були максимально схожі, а предмети, що належать до різних кластерів, настільки різні, наскільки це можливо. Кластери ідентифікуються за допомогою заходів подібності. Ці подібні заходи включають відстань, зв'язок та інтенсивність. Різні заходи подібності можуть бути вибрані на основі даних або заявки. [1]

У нефазних кластеризаціях (також відомих як жорстке кластеризація) дані поділяються на окремі кластери, де кожна точка даних може належати лише до одного кластера. У нечіткому кластерингу точки даних можуть потенційно належати до декількох кластерів.

Рейтинги учасників призначаються для кожного пункту (тегів) даних. Ці членські оцінки вказують на ступінь, до якої точки даних відносяться до кожного кластера. Таким чином, точки на краю кластера з нижчими членами класів можуть бути в кластері меншою мірою, ніж точки в центрі кластера.

Одним з найпоширеніших алгоритмів нечіткого кластеризації є алгоритм Fuzzy C-means clustering (FCM).

Кластеризація нечітких c-засобів (FCM) була розроблена Дж. К. Данном у 1973 р. [2] і поліпшена Дж.К.Бездеком в 1981 р. [3]

Алгоритм нечіткого c-means дуже схожий на алгоритм k-means:

1. Виберіть декілька кластерів.

2. Призначати коефіцієнти випадковим чином для кожної точки даних для перебування в кластері.

3. Повторюйте, доки алгоритм не згорнувся (тобто зміна коефіцієнтів між двома ітераціями не більше ніж задана порогова чутливість):

3.1. Обчислити центроїд для кожного кластера (показано нижче).

3.2. Для кожної точки даних обчислюйте його коефіцієнти буття в кластері. Центроїд - будь-яка точка x має набір коефіцієнтів, що дають ступінь буття в *k*-му кластері *wk(x)*. При нечітких с-засобах центроїд кластера є середнім з усіх точок, зважених за ступенем приналежності до кластера (формула 2.1).

(2.1)

$$c_k = \frac{\sum_x w_k(x)^m x}{\sum_x w_k(x)^m},$$

Де *m* - це гіперпараметр, який контролює, наскільки нечітким буде кластер. Чим вище це, тим краще це кластер буде в кінці.

Кластерні проблеми мають застосування в біології, медицині, психології, економіці та багатьох інших дисциплінах [6].

У галузі біоінформатики кластеризація використовується для ряду додатків. Одне використання - це метод розпізнавання образів для аналізу даних про експресію генів з мікрочипів або інших технологій [7]. У цьому випадку гени з подібними паттернами вираження згруповані в один і той же кластер, а різні кластери демонструють чіткі, добре розділені шаблони вираження. Використання кластеризації може дати уявлення про функцію та регуляцію генів [6]. Оскільки нечітка кластеризація дозволяє генам належать до декількох кластерів, це дозволяє ідентифікувати гени, які умовно корегулюються або спів-виражені [8]. Наприклад, до одного гена може діяти більше ніж один фактор транскрипції, і один ген може кодувати білок, який має більше ніж одну функцію. Таким чином, нечітка кластеризація є більш доцільною, ніж тверда кластеризація.

Нечіткі с-засоби є дуже важливим інструментом обробки зображень у кластеризації об'єктів на зображенні. У 70-х роках математики представили просторовий термін у алгоритм FCM, щоб підвищити точність кластеризації під шум [9]. Крім того, алгоритми FCM були використані для розмежування різних дій, використовуючи функції на основі зображення, такі як моменти Hu та Zernike [10]. Альтернативно, модель нечіткої логіки може бути описана на нечітких множинах, які визначаються на трьох компонентах кольорового простору HSL HSL та HSV;

Функції членства мають на меті описи кольорів, що відповідають людській інтуїції ідентифікації кольорів [11].

Сегментація зображень за допомогою алгоритмів кластеризації k-means вже давно використовується для розпізнавання образів, виявлення об'єктів та медичної візуалізації. Проте через обмеження в реальному світі, такі як шум, тінь та відмінності в камерах, традиційні жорсткі кластери часто не можуть надійно виконувати завдання обробки зображень, як зазначено вище. [13] Нечіткі кластери були запропоновані як більш відповідний алгоритм виконання цих завдань. Дано зображення сірого кольору, яке зазнало нечіткої кластеризації в Matlab. [14] Початкове зображення відображається поряд із кластеризованим зображенням. Кольори використовуються для візуального представлення трьох різних кластерів, які використовуються для визначення членства в кожному пікселі. Нижче наведено діаграму, яка визначає нечіткі коефіцієнти членності їх відповідних значень інтенсивності.

Під час інтелектуального групування файлів і веб-сайтів кластери можуть використовуватися для створення більш релевантного набору результатів пошуку порівняно з звичайними пошуковими системами, такими як Google. В даний час існує безліч веб-інструментів кластеризації, таких як Clusty. Це також може бути використано для повернення більш повного набору результатів у випадках, коли пошуковий термін може посилатися на вельми різні речі. Кожне чітке використання терміну відповідає унікальному кластері результатів, що дозволяє алгоритму рейтингу повертати загальні результати, вибравши найвищий результат з кожного кластера [47].

Враховуючи доступність технології та необхідність в універсальності системи щодо різнотипності класів інформації яку система повинна зберігати, то найбільш доцільним було б використання машинного навчання, проте цей метод потребував би етап налаштування та навчання нейронної мережі для розпізнавання класів.

Але планується, що динамічний реєстр будуть формувати самі користувачі. Тому, було обрано метод класифікації за бібліотекарем з доданням деяких рубрик у мета-інформацію посилання на інформаційний ресурс самою системою.

3. АНАЛІЗ ІСНУЮЧИХ ПРОГРАМНИХ СИСТЕМ ДЛЯ РОБОТИ З ДИНАМІЧНИМИ РЕЄСТРАМИ

На сьогоднішній день існує велика кількість програмних систем для налаштування динамічних реєстрів, але всі вони спеціалізуються на різних специфічних типах інформаційних ресурсів.

3.1 Веб-переглядач Google Chrome

Google Chrome (відомий просто як Chrome) - безкоштовний веб-браузер, розроблений компанією Google LLC [13]. Він був вперше випущений 2 вересня 2008 року для Microsoft Windows, і пізніше був перенесений на Linux, macOS, iOS та Android. Google Chrome також є основним компонентом ОС Chrome, де він слугує платформою для роботи веб-програм.

Google випускає більшість вихідних кодів Chrome як проект із відкритим вихідним кодом Chromium; [14] [15], проте сам Chrome є фірмовим програмним забезпеченням. [16] [13] Один компонент, який не є відкритим вихідним кодом, - це вбудований Adobe Flash Player (за замовчуванням Chrome відключений з вересня 2016 року [17]). Chrome використовував двигун макета WebKit до версії 27. Нова архітектура насамперед визнає той факт, що на сьогодні більшість веб-сайтів є не просто веб-сторінками, але веб-програмами. Заявленими перевагами цієї архітектури є підвищена стабільність, швидкість, безпека, а також чистий, простий та ефективний інтерфейс користувача (рисунок 3.1). Починаючи з версії 28, всі порти Chrome, крім порту iOS, використовують Blink, форк двигуна WebKit. [18] [19] [20]

Станом на 2018 рік, Google Chrome має 68% загальносвітової частки веб-браузерів як настільного браузера [21]. Вона також має 61% ринку на всіх платформах [22], оскільки вона має більше 50% частки на смартфонах; і тому Chrome найпопулярніший браузер практично у всіх країнах (більшість винятків в Африці). [23] Його успіх призвів до того, що Google розширює бренд "Chrome" на інших

продуктах, таких як ОС Chrome, Chromecast, Chromebook, Chromebit, Chromebox та Chromebase.

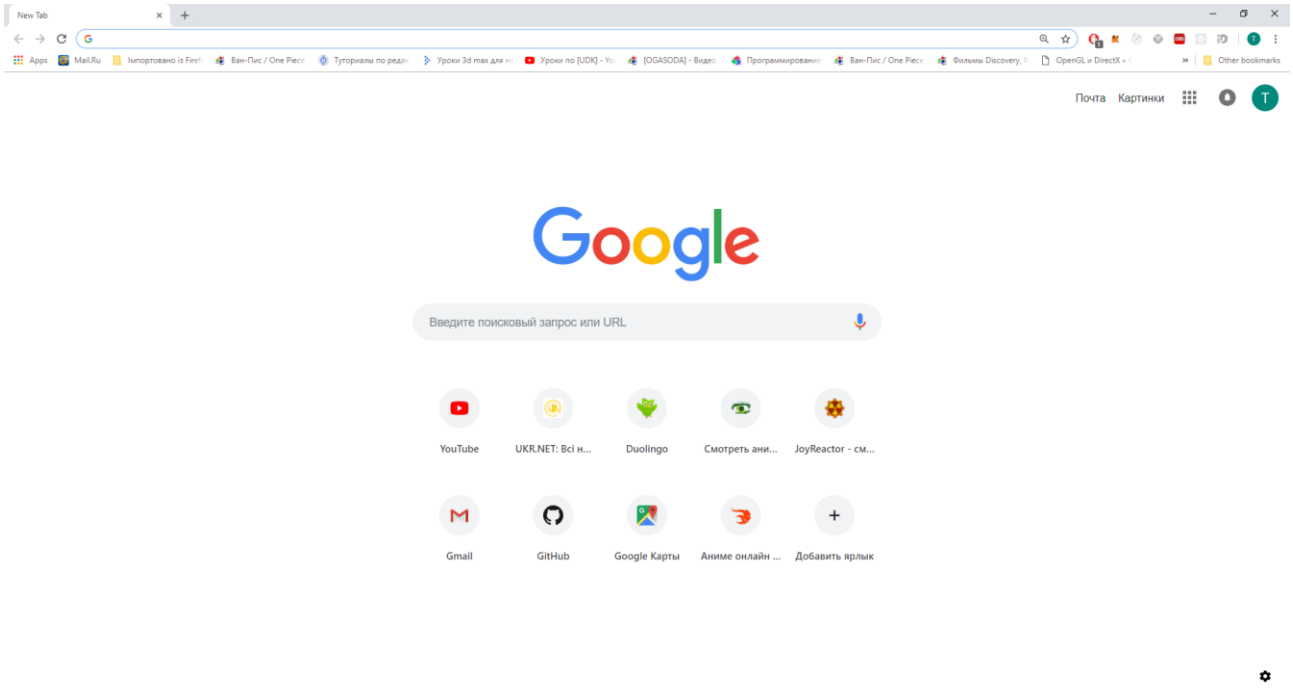

Рисунок 3.1 – Головна форма Google Chrome

На відміну від багатьох інших браузерів, в Chrome кожна вкладка є окремим процесом. У випадку, якщо процес обробки вмісту вкладки зависне, його можна буде закрити без ризику втратити всі дані в браузері. Панель вкладок є основним компонентом інтерфейсу і знаходиться у верхній частині вікна, а не під рядком адреси. Ця зміна виглядає досить контрастно в порівнянні з багатьма іншими браузерами. Вкладка легким перетягуванням на вільне місце може бути перенесена в окреме вікно (можливий також і зворотний варіант). Кожна вкладка має свій власний набір елементів, у тому числі Omnibox.

Omnibox — це адресний рядок, який знаходиться зверху кожної вкладки, він поєднує в собі адресний рядок і пошуковий рядок. Omnibox перенаправляє запит в пошукову систему в тому випадку, якщо адреса не відповідає правилам написання URL — наприклад, не містить точок, імені протоколу, косих рис, містить прогалини на початку адреси тощо. При введенні пошукового запиту Google, використовуючи функцію пошуку в Інтернеті «Мені пощастить», пропонує першу за списком URL-адресу (наприклад, при введенні в адресний рядок слова «Яндекс» браузер

автоматично запропонує варіант «www.yandex.ru»). Також автодоповнення пропонується пошуком по закладках та історії раніше відвіданих сторінок. У браузері можливий виклик різних пошукових машин прямо з адресного рядка. Для цього треба ввести скорочення для даної пошукової машини (наприклад, букву 'g' для google), потім натиснути клавішу «Пробіл» і ввести пошуковий запит. Скорочення для пошукових машин можна задавати самостійно. При відвідуванні сайтів браузер автоматично копіює звідти рядок пошукових запитів і наступного разу при введенні цього сайту автоматично буде запропоновано пошук з використання його пошукової системи.

Впродовж трьох років роботи Pwn2Own з 2009-2011 рр. Ніяких вразливостей у Chrome не було використано.

У Pwn2Own 2012 Chrome переміг французька команда, яка використовувала нульовий день у версії Flash, доставленому з Chrome, щоб повністю контролювати повністю виправлений 64-розрядний ПК Windows 7, використовуючи веб-сайт із блокуванням блокування, який подолав пісочниці Chrome.

Chrome була скомпрометована двічі на 2012 році CanSecWest Pwnium. Офіційну відповідь Google на експлуатацію доставив Джейсон Керсі, який привітав дослідників із зазначенням: "Ми також вважаємо, що обидва подання є творами мистецтва і заслуговують ширшого обміну та визнання". Виправлення для цих вразливостей було розгорнуто протягом 10 годин після подання.

Значна кількість вразливостей у системі Chrome у веб-переглядачі Adobe Flash Player. Наприклад, успішна атака Pwn2Own у 2016 році на Chrome спиралася на чотири вразливості системи безпеки. Дві з уразливостей були у Flash, одна - в Chrome, одна - в ядрі Windows. У 2016 році Google оголосив, що планує відмовитися від використання Flash Player у Chrome, починаючи з версії 53. Перший етап плану - відключити Flash для реклами та "фонової аналітики", з кінцевою метою повністю відключити його від наприкінці року, за винятком окремих сайтів, які, на думку Google, вважаються розбитими без нього. Після цього Flash буде відновлено, за винятком об'яв та фонової аналітики на основі кожного сайту.

Випущені документи, опубліковані WikiLeaks, під кодовою назвою Vault 7 і датовані 2013-2016 рр., Детально розповідають про можливості ЦРУ, такі як можливість компрометації веб-переглядачів (включаючи Google Chrome).

Переваги:

— висока швидкість роботи;

— великі можливості, які дозволяють виконувати будь-які операції створення і обробки реєстрів представленних у вигляді вкладок;

— великий набір команд фільтрації, за допомогою яких можна створювати найрізноманітніші збережені реєстри посилань.

3.2 Реєстр Windows

Реєстр Windows (англ. Windows Registry) — база даних, що зберігає параметри і налаштування для операційних систем Microsoft Windows 32-бітних версій, 64-бітних версій та Windows Mobile. Він містить інформацію і налаштування для всіх апаратних засобів, програмного забезпечення, користувачів тощо. Кожен раз, коли користувач змінює будь-які параметри в «Панелі керування», зміни відбуваються у реєстрі. (рисунок 3.2).

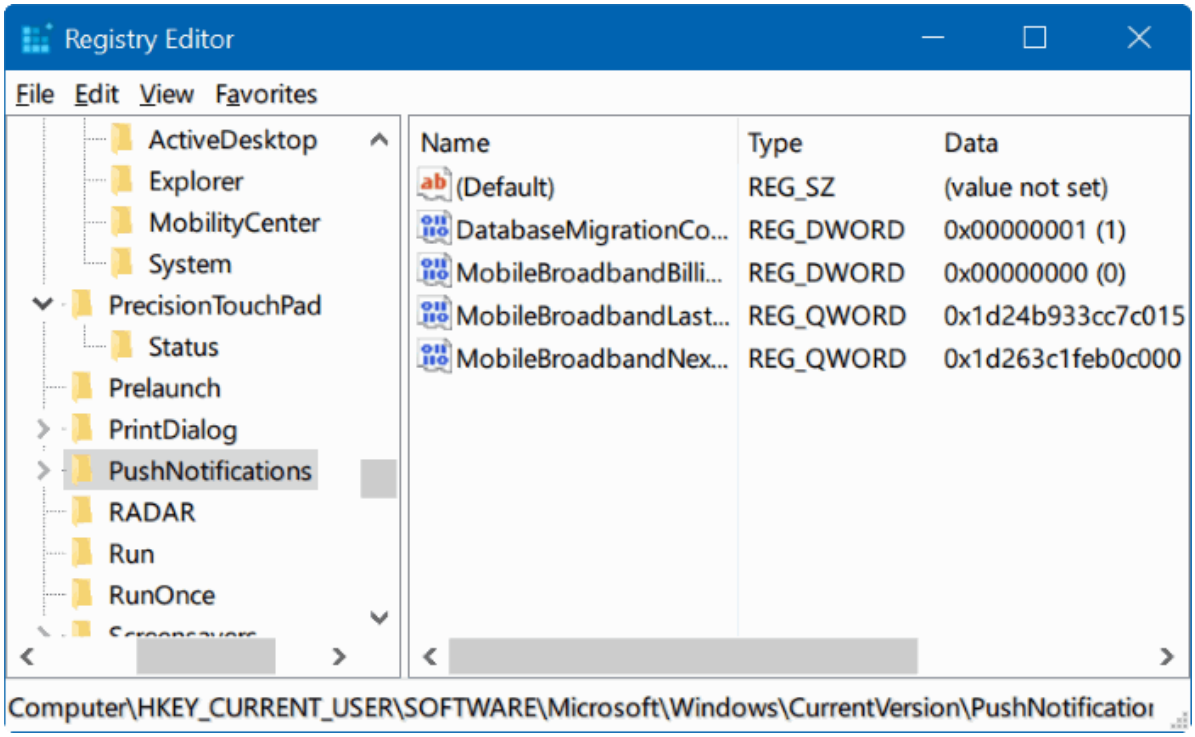

Рисунок 3.2 – Головна форма RegEdit

Реєстр Windows - це ієрархічна база даних, яка зберігає низькорівневі параметри для операційної системи Microsoft Windows та для програм, які використовують реєстр. Ядро, драйвери пристроїв, служби, менеджер облікових записів безпеки та користувацький інтерфейс можуть використовувати реєстр. Реєстр також забезпечує доступ до лічильників для профілювання продуктивності системи.

Для користувача, реєстр або реєстр Windows містить інформацію, налаштування, параметри та інші значення для програм та обладнання, встановлених у всіх версіях операційних систем Microsoft Windows. Наприклад, коли програма встановлена, всі нові розділи, що містять такі параметри, як розташування програми, його версія та спосіб запуску програми, додаються до реєстру Windows.

Після введення в Windows 3.1 реєстр Windows, в основному, зберігає інформацію про конфігурацію для компонентів на базі COM. Windows 95 та Windows NT розширили його використання, щоб раціоналізувати та централізувати інформацію у великій кількості файлів INI, які зберігали конфігурації для окремих програм і зберігалися в різних місцях. [1] [2] Для додатків Windows не потрібно використовувати реєстр Windows. Наприклад, додатки .NET Framework використовують файли XML для конфігурації, тоді як портативні програми зазвичай зберігають свої конфігураційні файли з їх виконуваними файлами.

До реєстру Windows, файли .INI зберігали налаштування кожної програми як текстовий файл, часто розміщувались в спільному розташуванні, де не було надано певних користувацьких налаштувань у сценарії для декількох користувачів. Навпаки, реєстр Windows зберігає всі параметри програми в одному логічному сховищі (крім ряду дискретних файлів) і в стандартній формі. На думку Microsoft, це має декілька переваг над файлами .INI. [2] [3] Оскільки аналіз файлів робиться набагато ефективніше за допомогою бінарного формату, його можна читати чи писати швидше, ніж файл INI. Крім того, сильно введені дані можуть зберігатися в реєстрі, на відміну від текстової інформації, що зберігається в файлах .INI. Це корисно при редагуванні клавіш вручну за допомогою RegEdit.exe, вбудованого редактора реєстру Windows. Оскільки встановлені користувачем параметри реєстру завантажуються з певного користувача, а не з місця розташування системи лише для читання, реєстр

дозволяє декільком користувачам поділитися однією машиною, а також дозволяє програмам працювати для менш привілейованих користувачів. Резервне копіювання та відновлення також спрощується, оскільки доступ до реєстру можна отримати через мережеве з'єднання для віддаленого керування / підтримки, у тому числі зі скриптів, за допомогою стандартного набору API, за умови, що служба віддаленого реєстру працює, і правила брандмауера дозволяють це зробити.

Оскільки реєстр є базою даних, він покращує цілісність системи за допомогою таких функцій, як атомні оновлення. Якщо два процеси намагаються одночасно оновлювати одне і те ж значення реєстру, одна зміна процесу буде передувати іншим, а загальна послідовність даних буде збережена. Якщо внесено зміни до файлів .INI, такі умови перегонів можуть призвести до невідповідності даних, які не збігаються з спробою оновлення. Windows Vista та пізніші операційні системи забезпечують транзакційні оновлення в реєстрі за допомогою диспетчера транзакцій ядра, розширюючи гарантії атомічності за допомогою кількох ключових та / або змінених значень з традиційною семантикою commit-abort.

Реєстр Windows було введено, щоб відмовитись від використання файлів INI, що використовувалися для збереження параметрів конфігурації програм Windows раніше (тобто кожна програма зберігала свої налаштування в окремому файлі). Тому ці файли мали тенденцію бути розкиданими по всій системі, що утруднювало спостереження і контроль за ними.

Ключ реєстру — це група розділів, підрозділів і параметрів реєстру, з якою пов'язано групу допоміжних файлів, де містяться резервні копії всіх цих даних. У Windows NT 4.0, Windows 2000, Windows XP, Windows Server 2003 та Windows Vista допоміжні файли для всіх ключів, окрім HKEY_CURRENT_USER, містяться в папці %SystemRoot%\System32\Config. Допоміжні файли для HKEY_CURRENT_USER розташовані в папці %SystemRoot%\Profiles\Username.

Реєстр в тому вигляді, як його використовує Windows і як бачить його користувач в процесі використання програм роботи з реєстром, деяким чином «ніде не зберігається». Щоб вийшло те, що бачить користувач, коли редагує реєстр, відбувається наступне:

- спочатку, в процесі установки і налаштування Windows, на диску формуються файли, в яких зберігається частина даних щодо конфігурації системи.

- потім, в процесі кожного завантаження системи, а так само в процесі кожного входу і виходу кожного з користувачів, формується якась віртуальна сутність, що називається «реєстром». Дані для формування «реєстру» беруться з тих самих файлів і з інших місць.

Файли .REG (також відомі як записи реєстру) - це текстові файли, які можна читати, для експорту та імпортування частин реєстру. У Windows 2000 та пізніших версіях вони містять початкову версію реєстру Windows Registry Version 5.00 на основі Unicode. У системах Windows 9x і NT 4.0 вони містять рядок REGEDIT4 і базуються на ANSI [18]. Файли формату Windows 9x.REG сумісні з Windows 2000 та пізнішими версіями. Редактор реєстру Windows на цих системах також підтримує експорт файлів .REG у форматі Windows 9x / NT.

3.3. Пошукова система Google

Пошукова система Google, також називається Google Web Search або просто Google, - це веб-пошукова система, розроблена компанією Google LLC. Це найпопулярніша пошукова система на Всесвітній павутині на всіх платформах, що на 92,3% ринку станом на вересень 2018 року [5] обробляє більше трьох мільярдів пошуків щодня [6].

Порядок результатів пошуку, отриманих компанією Google, базується частково на системі пріоритетних рейтингів, що називається "PageRank". Пошукова система Google також пропонує різні варіанти персоналізованого пошуку, використовуючи символи для включення, виключення, вказування або вимагання певної поведінки в пошукових системах, а також пропонує спеціалізовані інтерактивні враження, такі як стан польоту та відстеження пакетів, прогнози погоди, валюта, одиниця і час конверсії, слово визначення тощо.

Основною метою пошуку Google є полювання на текст у публічно доступних документах, що пропонуються веб-серверами, на відміну від інших даних, таких як

зображення або дані, що містяться в базах даних. Його спочатку розробляли Ларрі Пейдж і Сергій Брін в 1997 році. [3] У червні 2011 року компанія Google представила "Пошук Google Voice", щоб шукати слова, а не набрані словами [7]. У травні 2012 року компанія Google представила функцію пошуку семантичного пошуку знань у США (рисунок 3.3).

Аналіз частоти пошукових термінів може вказувати на економічні, соціальні та медичні тенденції [8]. Дані щодо періодичності використання пошукових термінів у Google можна відкрито проаналізувати через Google Trends, і вони підтверджують, що вони співвідносяться зі спалахом грипу та рівнем безробіття та надають інформацію швидше, ніж традиційні методи та опитування. Станом на середину 2016 року пошукова система Google почала спиратися на глибокі нейронні мережі [9].

Конкуренти Google включають Baidu та Soso.com у Китаї; Naver.com і Daum.net в Південній Кореї; Яндекс у Росії; Seznam.cz в Чеській Республіці; Yahoo в Японії, Тайвані та США, а також Bing та DuckDuckGo. [10] Деякі менші пошукові системи пропонують об'єкти, недоступні в Google, наприклад не зберігати будь-яку приватну або відстежувану інформацію.

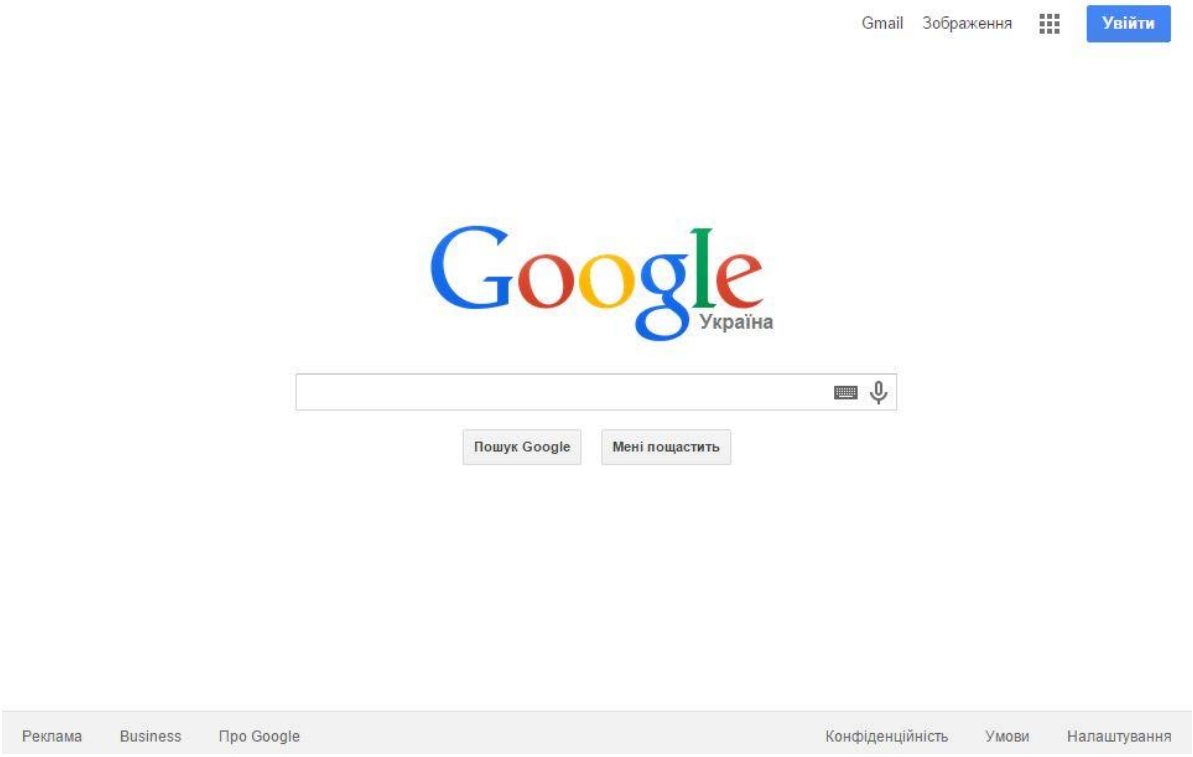

Рисунок 3.3 – Головна форма Google Україна

Інтерфейс Google містить досить складну мову запитів, що дозволяє обмежити область пошуку окремими доменами, мовами, типами файлів тощо. Наприклад, пошук «intitle: Google site: wikipedia.org» видасть всі статті Вікіпедії всіма мовами, в заголовку яких зустрічається слово «Google» [5]. Потужна мова запитів у руках хакерів може бути використана для дослідження веб-сайтів на вразливість.

Підвищення успіху Google зумовлено, в основному, запатентованим алгоритмом, який називається PageRank, який допомагає класифікувати веб-сторінки, які відповідають даному рядку пошуку. [11] Коли Google був Стенфордським дослідницьким проектом, його називали BackRub, тому що технологія перевіряє зворотні посилання, щоб визначити важливість сайту. Попередні на основі ключових слів методи ранжирування результатів пошуку, що використовуються багатьма пошуковими системами, які були ще раз популярнішими, ніж Google, оцінять сторінки, як часто пошукові терміни траплялись на сторінці або наскільки сильно асоціювалися пошукові терміни з кожною отриманою сторінкою. Замість цього алгоритм PageRank аналізує людські генеровані посилання, якщо веб-сторінки, зв'язані з багатьох важливих сторінок, можуть бути важливими. Алгоритм обчислює рекурсивну оцінку для сторінок, виходячи із зваженої суми PageRanks сторінок, що з ними зв'язуються. Як видається, PageRank добре співвідноситься з поняттями важливості людини. Окрім PageRank, Google з часом додала ще багато інших секретних критеріїв для визначення рейтингу сторінок у списках результатів. Повідомляється, що це понад 250 різних показників [12] [13], специфіка яких зберігається в секреті, щоб уникнути труднощів, сформованих мошенниками, і допомогти Google зберігати перевагу над своїми конкурентами у всьому світі.

У потенційному натяку на майбутнє напрямку Google для свого алгоритму пошуку Ерік Шмідт, тодішній виконавчий директор Google, заявив в інтерв'ю Financial Times в 2007 році: "Мета полягає в тому, щоб дозволити користувачам Google мати можливість задавати таке питання, як" Що чи повинен я робити завтра? і "Яку роботу я візьму?" [14]. Шмідт підтвердив це під час інтерв'ю з Wall Street Journal 2010: "Я

думаю, що більшість людей не хочуть, щоб Google відповідав на їхні запитання, вони хочуть, щоб Google розповідав їм, що вони повинні робити далі". [15].

У 2013 році Європейська Комісія встановила, що Пошук Google віддавав перевагу власним продуктам Google, замість того, щоб пропонувати споживачам найкращий результат для їхніх потреб [16]. У лютому 2015 року компанія Google оголосила про серйозні зміни у своєму алгоритмі мобільного пошуку, який буде сприяти мобільному дружнім за інших веб-сайтах. Близько 60% трафіку пошуку в мережі Google надходять із мобільних телефонів. Google каже, що хоче, щоб її користувачі мали доступ до веб-сайтів високої якості. Ті сайти, у яких бракує дружнього інтерфейсу для мобільних пристроїв, будуть знижені, і очікується, що це оновлення призведе до руйнування ряду. Підприємства, які не змогли відповідно оновити свої веб-сайти, можуть побачити падіння в їх регулярному трафіку веб-сайтів. [17]

Google показує сотні терабайт інформації з веб-сторінок. [18] Для веб-сайтів, які наразі відсутні або іншим чином недоступні, Google надає посилання на кешовані версії сайту, створені за останніми індексацією пошукової системи цієї сторінки. [19] Крім того, Google індексує деякі типи файлів, які можуть відображати PDF-файли користувачам, документи Word, електронні таблиці Excel, презентації PowerPoint, певний мультимедійний вміст Flash та прості текстові файли [20]. Користувачі також можуть активувати "Безпечний пошук", технологію фільтрації, спрямовану на запобігання показу явного та порнографічного вмісту в результатах пошуку [21].

У 2012 році Google змінив інструменти індексації пошуку, щоб знищити сайти, які були звинувачені в піратстві [22].

Оскільки Google є найпопулярнішою пошуковою системою, багато веб-майстрів намагаються впливати на рейтинги Google на своєму веб-сайті. Індустрія консультантів виникла, щоб допомогти веб-сайтам підвищити свої рейтинги в Google та інших пошукових системах. Це поле, що називається оптимізація пошукових систем, намагається розрізнити шаблони в списках пошукових систем, а потім розробляти методологію для покращення рейтингу, щоб залучити більшу кількість пошуків на сайти своїх клієнтів. Пошукова оптимізація охоплює як фактори "на сторінці" (наприклад, копіювання тіла, елементи заголовка, елементи заголовка H1,

так і значення атрибута зображення) та фактори оптимізації Off Page (наприклад, текст прив'язки та PageRank).

Загальна ідея полягає в тому, щоб впливати на алгоритм відповідності Google, включивши ключові слова, націлені на різні місця "на сторінці", зокрема елемент заголовка та копію тіла (примітка: вище вгорі сторінки, імовірно, чим краще його ключове слово і, таким чином, краще рейтинг) Однак занадто багато випадків ключового слова призводять до того, що сторінка виглядає підозрілою в алгоритмах перевірки спаму Google. Google опублікувала рекомендації для власників веб-сайтів, які хотіли б підвищити свої рейтинги при використанні законних консультантів з оптимізації [25]. Було висунуто гіпотезу, ймовірно, є думка власника одного з підприємств, про які мали місце численні скарги, що негативна реклама, наприклад, численні скарги споживачів, може також сприяти підвищенню рейтингу сторінок в Пошуку Google як сприятливі коментарі. [26] Особлива проблема, що розглядається в статті The New York Times, яка включала в себе DecorMyEyes, була опублікована незабаром після нерозкритих виправлень в алгоритмі Google. Згідно з даними Google, не часто публікувалися скарги споживачів на DecorMyEyes, які призвели до високого рейтингу, але згадували на новинних веб-сайтах події, які вплинули на фірму, такі як судові дії проти неї. Пошукова консоль Google допомагає перевіряти веб-сайти, які використовують дубльований чи захищений вміст. [27]

Компанія "Google" запустила "універсальний пошук" 16 травня 2007 р. Як ідею, яка об'єднала результати з різних типів пошукових систем у один. До загального пошуку стандартний пошук Google складатиметься лише з посилань на веб-сайти. Універсальний пошук, однак, включає в себе широкий спектр джерел, включаючи веб-сайти, новини, картинки, карти, блоги, відео та багато іншого, усі вони відображаються на тій же сторінці результатів пошуку. [28] [29] Марісса Майєр, тодішній віце-президент з пошукових продуктів і досвіду роботи, описав ціль універсального пошуку як "ми намагаємося розірвати стіни, які традиційно відокремлювали наші різні властивості пошуку та інтегрували величезну кількість доступної інформації в один простий набір результатів пошуку. [30]

У серпні 2009 року Google закликав веб-розробників тестувати нову архітектуру пошуку, з кодованим назвою "Caffeine", і надати свої відгуки. Нова архітектура не забезпечила ніяких візуальних відмінностей у користувальницькому інтерфейсі, але додав значні покращення швидкості та нову інфраструктуру індексації "під капотом". Цей крок був інтерпретований у деяких розділах, як відповідь на недавній випуск Microsoft відновленої версії власної служби пошуку, перейменованого в Bing, а також запуск Wolfram Alpha, нової пошукової системи на основі "обчислювальних знань" [31]. [32] Компанія Google оголосила про завершення роботи "Кофеїну" 8 червня 2010 року, вимагаючи 50% новітніх результатів через постійне оновлення його індексу [33].

За допомогою " Caffeine" Google перемістив свою систему зворотної індексації з MapReduce і на платформу розподіленої бази даних компанії Bigtable [34] [35].

Графік знань є базою знань, що використовується компанією Google для покращення результатів пошукової системи з інформацією, зібраною з різних джерел [36]. Ця інформація представлена користувачам у вікні праворуч від результатів пошуку [37]. У травні 2012 року [36] у пошукових системах Google додано графіки знань, що починаються в США, а міжнародна експансія до кінця року [38]. Інформація, охоплена Графіком знань, значно зросла після запуску, тричі збільшивши її початковий розмір протягом семи місяців [39] і зможучи відповісти "приблизно на третину" із 100 мільярдів пошукових запитів щомісяця, оброблених Google у травні 2016 року. [40] Ця інформація часто використовується як розмовна відповідь у запитах Google Assistant [41] та Google Home [42]. Графік знань був підданий критиці за надання відповідей без джерельної атрибуції [40].

У 2013 році Google суттєво покращив свій алгоритм пошуку з "Hummingbird". Її назва походить від швидкості і точності колібрі. [43] Ця зміна була оголошена 26 вересня 2013 року, вже щомісяця була використана. [44] "Колібниця" приділяє більше уваги питанням природної мови, розглядаючи контекст та значення над окремими ключовими словами [43]. Він також поглиблює вміст на окремих сторінках веб-сайту, з покращеною здатністю вести користувачів безпосередньо на найбільш відповідну сторінку, а не просто на головну сторінку веб-сайту. [45] Оновлення ознаменувало

найважливішу зміну в пошуку Google впродовж багатьох років, з більш "людськими" пошуковими взаємодіями [46] та значно більшим зосередженням на розмовах та змісті [43]. Таким чином, веб-розробникам та письменникам було запропоновано оптимізувати свої сайти за допомогою натурального написання, а не примусових ключових слів, а також ефективно використовувати технічні веб-розробки для навігації на місці [47].

4. ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

Система розроблена з використанням мови програмування JavaScript для клієнтської частини та NodeJs для серверної. В клієнтській частині: користувацький інтерфейс розроблений за допомогою фреймворку React та CSS FlexBox, а взаємодія із серверною частиною відбувається з використанням REST API та Fetch API. В серверній частині використано фреймворк Express і налаштовано роботи через REST API.

В даній роботі, у реалізованому програмному комплексі, користувачу доступні частотні та атрибутивні ознаки динамічності для встановлення та редагування, а саме:

- Реєстрація частоти запитів щодо отримання мета-інформації певного ресурсу;
- Додавання нових параметрів мета-інформації до існуючих посилань на електронні інформаційні ресурси;
- Додавання нових посилань на електронні інформаційні ресурси;
- Додавання нового типу графічного відображення реєстру для користувача;

Проте серверна складова програмного комплексу для керування динамічним реєстром може підтримувати також темпоральну динамічність реєстрів та параметрів електронних інформаційних ресурсів, наприклад:

- Степінь важливості інформаційного ресурсу;
- Степінь секретності інформаційного ресурсу;
- Степінь доступу до інформаційного ресурсу;
- Актуальність інформації у певного електронному інформаційному ресурсі;

4.1. Особливості платформи JavaScript та NodeJs

Мова Javascript — динамічна, об'єктно-орієнтована[6] прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв веб-сторінок, що надає можливість на стороні клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно

обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб-сторінки.

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Незважаючи на схожість назв, мови Java та JavaScript є двома різними мовами, що мають відмінну семантику, хоча й мають схожі риси в стандартних бібліотеках та правилах іменування. Синтаксис обох мов отриманий «у спадок» від мови C, але семантика та дизайн JavaScript є результатом впливу мов Self та Scheme.[7]

Node.js — платформа з відкритим кодом для виконання високопродуктивних мережевих застосунків, написаних мовою JavaScript. Засновником платформи є Раян Дал (Ryan Dahl). Якщо раніше Javascript застосовувався для обробки даних в браузері на сторонні користувача, то node.js надав можливість виконувати JavaScript-скрипти на сервері та відправляти користувачеві результат їх виконання. Платформа Node.js перетворила JavaScript на мову загального використання з великою спільнотою розробників.

Для забезпечення обробки великої кількості паралельних запитів у Node.js використовується асинхронна модель запуску коду, заснована на обробці подій в неблокуючому режимі та визначенні обробників зворотніх викликів (callback). Як способи мультиплексування з'єднань підтримується epoll, kqueue, /dev/poll і select. Для мультиплексування з'єднань використовується бібліотека libuv, для створення пулу потоків (thread pool) задіяна бібліотека libeio, для виконання DNS-запитів у неблокуючому режимі інтегрований c-ares. Всі системні виклики, що спричиняють блокування, виконуються всередині пула потоків і потім, як і обробники сигналів, передають результат своєї роботи назад через неіменовані канали (pipe).

За своєю суттю Node.js схожий на фреймворки Perl AnyEvent, Ruby Event Machine і Python Twisted, але цикл обробки подій (event loop) у Node.js прихований від

розробника і нагадує обробку подій у веб-застосунку, що працює в браузері. При написанні програм для Node.js необхідно враховувати специфіку подієво-орієнтованого програмування.

4.2. Фреймворки React та Express

React (старі назви: React.js, ReactJS) — відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту веб-сторінки, з якими стикаються в розробці односторінкових застосунків. Розробляється Facebook, Instagram і спільнотою індивідуальних розробників.

React не намагається надати повну "схему додатків". Він розроблений спеціально для побудови користувацьких інтерфейсів [3], і тому не включає в себе безліч інструментів, які деякі розробники можуть вважати необхідними для створення програми. Це дозволяє вибрати будь-які бібліотеки, які розробник вважає за краще виконувати такі завдання, як здійснення доступу до мережі або локальне зберігання даних. Загальні закономірності використання з'являються, коли бібліотека дозріває.

React дозволяє розробникам створювати великі веб-застосунки, які використовують дані, котрі змінюються з часом, без перезавантаження сторінки. Його мета полягає в тому, щоб бути швидким, простим, масштабованим. React обробляє тільки користувацький інтерфейс у застосунках. Це відповідає видові у шаблоні модель-вид-контролер (MVC), і може бути використане у поєднанні з іншими JavaScript бібліотеками або в великих фреймворках MVC, таких як AngularJS[8]. Він також може бути використаний з React на основі надбудов, щоб піклуватися про частини без користувацького інтерфейсу побудови веб-застосунків. Як бібліотеку інтерфейсу користувача React найчастіше використовують разом з іншими бібліотеками, такими як Redux.

Ще однією помітною особливістю є використання "віртуальної моделі об'єктів документів" або "віртуального DOM". React створює кеш-пам'ять даних у пам'яті, обчислює отримані відмінності, а потім оновлює відображений DOM веб-переглядач ефективно. [14] Це дозволяє програмісту писати код так, ніби вся сторінка

відображається на кожну зміну, а бібліотеки React відтворюють тільки ті компоненти, які дійсно змінюються.

В даний час React використовують Khan Academy, Netflix, Yahoo, Airbnb, Sony, Atlassian та інші.

Express.js, або просто Express — програмний каркас розробки веб-застосунків для Node.js, реалізований як вільне і відкрите програмне забезпечення під ліцензією MIT. Він спроектований для створення веб-застосунків і API[9]. Де-факто є стандартним каркасом для Node.js. Автор фреймворка, TJ Holowaychuk, описує його як створений на основі написаного на мові Ruby каркаса Sinatra, маючи на увазі, що він мінімалістичний, але має велику кількість плаґінів, шо підключаються.

Express є бекендом для програмного стека MEAN, разом з базою даних MongoDB і каркасом AngularJS для фронтенду.

4.3. Особливості архітектури REST API

Підхід Representational State Transfer (REST) - це програмний архітектурний стиль, який визначає сукупність обмежень, які слід використовувати для створення веб-служб. Веб-сервіси, які відповідають архітектурному стилю REST, називаються RESTful web services, забезпечують сумісність між комп'ютерними системами в Інтернеті. RESTful веб-сервіси дозволяють запитуючим системам отримувати доступ до текстуальних уявлень веб-ресурсів і керувати ними, використовуючи єдиний та попередньо визначений набір операцій без громадянства. Інші види веб-сервісів, наприклад веб-служби SOAP, виставляють власні довільні набори операцій. [1]

"Веб-ресурси" вперше були визначені в World Wide Web як документи або файли, визначені їх URL-адресами. Однак сьогодні вони мають набагато більш загальне і абстрактне визначення, яке охоплює всі речі або об'єкти, які можуть бути ідентифіковані, названі, адресовані або оброблені будь-яким способом в Інтернеті. У RESTful веб-службі запити, зроблені на URI ресурсу, викликають відповідь із корисним завантаженням, відформатованим у HTML, XML, JSON або деякому іншому форматі. Відповідь може підтвердити, що деякі зміни були внесені в

збережений ресурс, а відповідь може забезпечити гіпертекстові посилання на інші пов'язані ресурси або збірки ресурсів. Коли HTTP використовується, як це найчастіше, доступні операції GET, POST, PUT, DELETE та інші попередньо визначені методи CRUD HTTP.

Використовуючи протокол безстанового характеру та стандартні операції, системи RESTful прагнуть до швидкої продуктивності, надійності та здатності зростати, повторно використовуючи компоненти, які можна керувати та оновлювати, не впливаючи на систему в цілому навіть під час роботи.

Термін «репрезентативний стан» був введений та визначений Рой Філдінг у докторській дисертації в 2000 році. [2] [3] Дирекція Філдінга пояснила принципи

REST, які були відомі як "об'єктна модель HTTP", починаючи з 1994 р., І використовувалися при розробці стандартів HTTP 1.1 та уніфікованих ідентифікаторів ресурсів (URI). [4] [5] [6] Термін призначений для виявлення того, як веде себе добре продумана веб-програма: це мережа веб-ресурсів (віртуальна машина-станція), де користувач проходить через додаток, вибираючи посилання, такі як / user / tom і такі операції, як GET або DELETE (переходи штату), внаслідок чого наступний ресурс (що представляє наступний стан програми) передається користувачеві для їх використання.

Обмеження архітектурного стилю REST впливають на наступні архітектурні властивості: [2] [8]

- продуктивність в компонентних взаємодій, що може бути домінуючим фактором продуктивності, сприйманої користувачем, та ефективності мережі; [9]

масштабованість, що дозволяє підтримувати велику кількість компонентів та взаємодію між компонентами. Рой Філдінг описує вплив REST на масштабованість наступним чином: "Відокремлення проблем від клієнт-сервер REST спрощує реалізацію компонентів, зменшує складність семантики з'єднувачів, підвищує ефективність настройки продуктивності та підвищує масштабованість чистих серверних компонентів. Шарові системи обмежень дозволяють посередникам-проксі-серверам, шлюзам і брандмауерам-вводити в різних точках зв'язку без зміни інтерфейсів між компонентами, тим самим дозволяючи їм допомагати в

комунікаційному перекладі або покращувати продуктивність за допомогою широкомасштабного спільного кешування. REST дозволяє здійснювати проміжну обробку за рахунок обмеження повідомлень самодописуючою: взаємодія безпосадових між запитами, стандартні методи та типи носіїв використовуються для позначення семантики та обміну інформацією, а відповіді явно вказують на кешування. [2] ";

- простота єдиного інтерфейсу;

- модифікованість компонентів для задоволення мінливих потреб (навіть якщо програма працює);

- видимість зв'язку між компонентами службовими агентами;

- переносимість компонентів шляхом переміщення програмного коду з даними;

- надійність у стійкості до аварії на рівні системи при наявності збоїв у компонентах, роз'ємах або даних [9].

Шість напрямних обмежень визначають систему RESTful. [8] [10] Ці обмеження обмежують способи, якими сервер може обробляти та реагувати на запити клієнта, таким чином, за допомогою роботи в рамках цих обмежень система отримує бажані нефункціональні властивості, такі як продуктивність, масштабованість, простота, модифікованість, видимість, портативність та надійність. [2] Якщо система порушує будь-які необхідні обмеження, її не можна вважати RESTful.

Офіційні обмеження REST такі:

1. Принцип клієнт-серверних обмежень полягає у поділі проблем. Розподіл проблем із інтерфейсом користувача з використанням сховищ даних покращує переносимість користувацького інтерфейсу на різних платформах. Це також покращує масштабованість, спрощуючи компоненти сервера. Однак, найважливішим для Інтернету є те, що поділ дозволяє компонентам еволюціонувати самостійно, тим самим підтримуючи потребу Інтернету в кількох організаційних областях [2].

2. Зв'язок між клієнтом і сервером обмежується тим, що клієнтський контекст зберігається на сервері між запитами. Кожний запит від будь-якого клієнта містить

всю інформацію, необхідну для обслуговування запиту, і стан сеансу зберігається в клієнті. Стан сеансу може передаватися сервером на іншу службу, таку як база даних, щоб підтримувати стійкий стан протягом певного періоду та дозволяти автентифікацію. Клієнт починає надсилати запити, коли він готовий перейти до нового стану. Хоча один або більше запитів є вичерпними, клієнт вважається перехідним. Представлення кожного стану програми містить посилання, які можуть бути використані в наступний раз, коли клієнт вирішить розпочати новий стан-перехід [11].

3. Як і в World Wide Web, клієнти та посередники можуть кешувати відповіді. Відповіді, таким чином, недвозначно або явним чином, визначають себе як кешовані або не дозволяють клієнтам отримувати застарілі або невідповідні дані у відповідь на подальші запити. Добре кероване кешування частково або повністю виключає деяку взаємодію між клієнтом і сервером, що дозволяє покращити масштабованість та продуктивність.

4. Клієнт не може звичайно визначити, чи він підключений безпосередньо до кінцевого сервера, чи до посередника на цьому шляху. Посередні сервери можуть покращувати масштабованість системи, увімкнувши балансування навантаження та надаючи загальні кеші. Вони також можуть застосовувати політику безпеки.

5. Сервери можуть тимчасово розширювати або налаштовувати функціональність клієнта, передаючи виконуваний код. Наприклад, складні компоненти, такі як аплети Java, та сценарії на стороні клієнта, такі як JavaScript.

6. Обмеження інтерфейсу є основоположним для розробки будь-якої системи RESTful [2]. Це спрощує і відокремлює архітектуру, яка дозволяє кожній частині розвиватися самостійно. Чотири обмеження для цього уніфікованого інтерфейсу:

6.1. Індивідуальні ресурси ідентифікуються в запитах, наприклад, використовуючи URI в RESTful Web-службах. Самі ресурси концептуально відокремлені від представлень, які повертаються клієнту. Наприклад, сервер може

надсилати дані з його бази даних як HTML, XML або як JSON, жоден з яких не є внутрішнім представленням сервера.

6.2. Коли клієнт має представлення ресурсу, включаючи будь-які додані метадані, він має достатньо інформації для зміни або видалення ресурсу.

6.3. Кожне повідомлення містить достатньо інформації для опису обробки повідомлення. Наприклад, який аналізатор для виклику може бути вказаний за типом носіїв. [2]

6.4. Отримавши доступ до початкового URI для програми REST, аналогічно тому, як веб-користувач має доступ до домашньої сторінки веб-сайту, клієнт REST повинен мати можливість динамічно користуватись послугами, наданими сервером, щоб виявити всі доступні дії та ресурси, які йому потрібні. Оскільки процес доступу надходить, сервер реагує на текст, який містить гіперпосилання на інші дії, які наразі доступні. Клієнту немає жорсткого кодування з інформацією щодо структури або динаміки застосування [12].

На відміну від Web-сервісів на базі SOAP, немає "офіційних" стандартів для RESTful Web API. Це тому, що REST є архітектурним стилем, а SOAP - протоколом. REST сам по собі не є стандартом, але реалізація RESTful використовує стандарти, такі як HTTP, URI, JSON та XML. Багато розробників також описують свої API як REST, навіть якщо ці API фактично не відповідають усім описаним вище архітектурним обмеженням (особливо рівномірному обмеженню інтерфейсу) [15].

4.4. Опис моделі даних системи

Програмний комплекс містить 6 серверних ендпоінтів та 2 класи (рисунок 5.1):

1. Ендпоінт Root (/) — шлях за замовчуванням. Відповідає за надання графічного інтерфейсу користувачу. Дозволяє лише зчитувати інформацію.

2. Ендпоінт Static (/static) — допоміжний шлях, який відкритий для публічного доступу і містить необхідні для клієнтської частини скрипти, стилі, HTML-файли, зображення та відео. Дозволяє лише зчитувати інформацію.

3. Ендпоінт API (/api) — шлях за замовчуванням для клієнтської логіки. Надає всі створені та збережені в локальній базі даних екземпляри класів Filter і File. Дозволяє лише зчитувати інформацію.

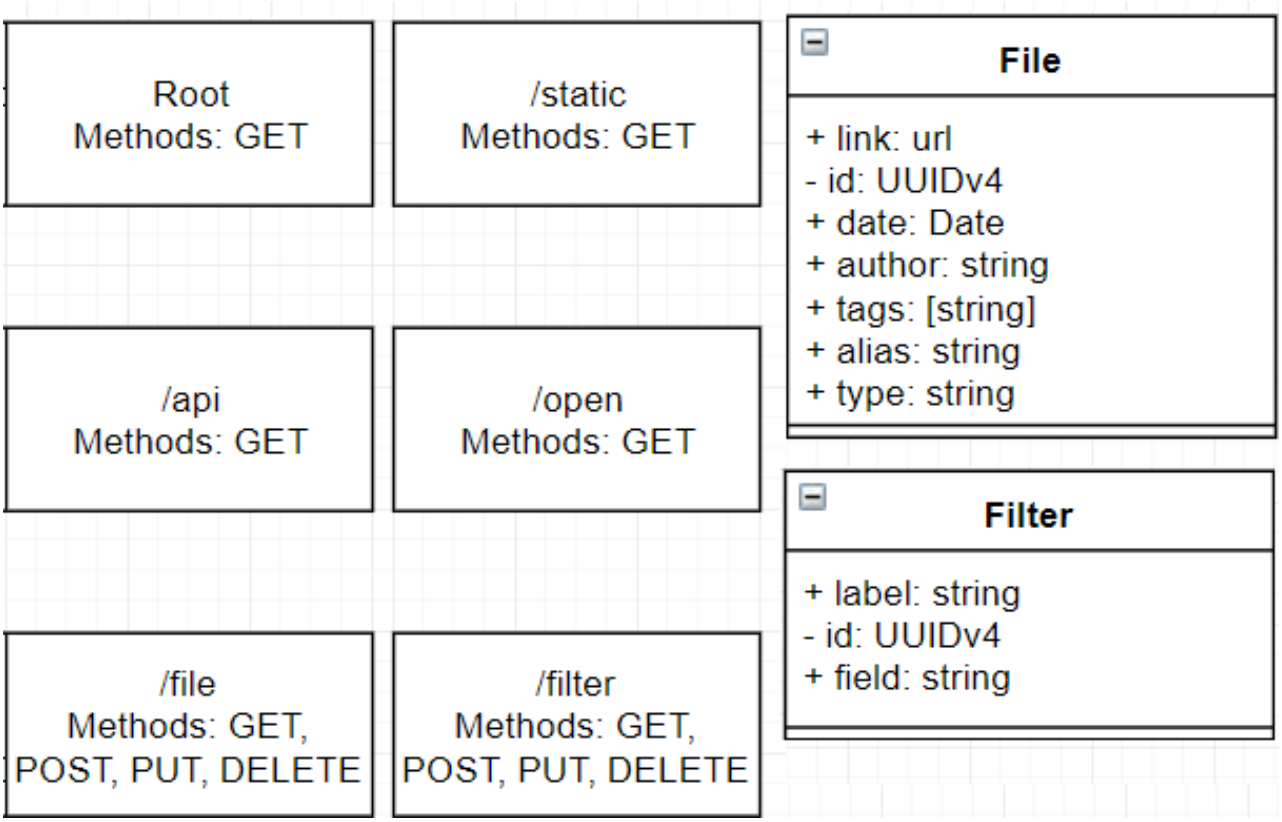

Рисунок 4.1 — Опис моделі даних

4. Ендпоінт Open (/open) — шлях для доступу до віддаленого доступу до операційної системи серверу. Дозволяє відкривати файли та посилання, що зберігаються у екземплярі класу File.

5. Ендпоінт File (/file) — шлях для доступу до операції над екземплярами класу File. Дозволяє створювати нові, редагувати існуючі та видаляти екземпляри класу File. Підтримує зчитування певного екземпляру через параметри запиту.

6. Ендпоінт Filter (/filter) — шлях для доступу до операцій над екземплярами класу Filter. Дозволяє створювати нові, редагувати існуючі та видаляти екземпляри класу Filter. Підтримує зчитування певного екземпляру через параметри запиту.

7. Клас File — клас, що містить мета-дані по певному посиланню (це може бути або посилання на віддалений електронний інформаційний ресурс або на локальний файл): саме посилання, дату створення, автора екземпляру, масив тегів, скорочену назву, тип посилання або файлу.

8. Клас Filter — допоміжний клас, який встановлює опис щодо фільтрації, групування та сортування певного класу або кластеру екземплярів класу File.

4.5. Опис компонентів клієнтської та серверної складової

Система розроблена за допомогою мови програмування JavaScript, а саме – клієнтська та серверна частини. Клієнтська частина виконана за допомогою фреймворку ReactJS, вихідний код якого потім перекомпілювано у виконуваний код JavaScript для використання веб-переглядачем. Клієнтська частина складається з графічного інтерфейсу для взаємодії з ресурсами динамічного реєстру – додавання посилань на нові ресурси та визначення для них частини мета-інформації. Розроблені функції клієнтської частини:

- Додавання нового посилання на електронний інформаційний ресурс;
- Редагування та перегляд існуючої мета-інформації певного ресурсу користувачем системи;
- Видалення посилання на інформаційний ресурс с динамічного реєстру (тобто із бази даних на серверній частині);
- Групування електронних інформаційних ресурсів за певними параметрами ресурсів у реєстрі, зокрема:
    1. За датою створення.
    2. За типом електронного інформаційного ресурсу.
    3. За автором ресурсу.
    4. За реєстратором ресурсу.
    5. За каталогами, в яких знаходиться відповідний електронний інформаційний ресурс.
    6. За назвою ресурсу.

- Сортування та пошук серед бази даних реєстру певних електронних інформаційних ресурсів (за параметрами, вказаними вище).

Серверна частина системи була створена за допомогою платформи NodeJS та фреймворку Express. Були розроблені наступні функції:

- Створення, редагування, видалення та отримання списку електронних інформаційних ресурсів у реєстрі за допомогою протоколу HTTP. Згідно клієнтської частини, кожен ресурс має наступні параметри:

1. Посилання на електронний інформаційний ресурс.
2. Дата створення посилання на інформаційний ресурс.
3. Реєстратор посилання на інформаційний ресурс.
4. Автор електронного інформаційного ресурсу.
5. Коротка назва посилання на інформаційний ресурс.
6. Тип електронного інформаційного ресурсу.
7. Відповідні каталоги, до яких відноситься електронний інформаційний ресурс.
8. Інші додаткові параметри ресурсу, які можна вказати через запит на сервер (мається на увазі, що усі попередні параметри можна вказати або через клієнтську частину або сервер встановить ці параметри автоматично).

- Створення, редагування, видалення та отримання списку фільтрів електронних інформаційних ресурсів.

- Надання доступу до реєстру (реєстрація, редагування, видалення та перегляд посилань на інформаційні ресурси) іншим сервісам через API, що може бути використано, наприклад, для створення системи адміністрації та контролю доступу до реєстру.

За паралельну та послідовну обробку запитів відповідає фреймворк Express, який працює згідно із архітектурою REST API(рисунок 5.2).

У випадку користуванням декількох клієнтів, запити виконуються асинхронно, використовуючи транзакційний доступ до локальної бази даних. Проте за синхронізацію та оновлення даних, клієнт повинен відповідати самостійно через

односторонню природу запитів (сервер не може сповіщати клієнтів під час зміни бази даних іншим клієнтом).

```
app.post('/file', function (req, res) {
    console.log("POST /file");
    appendDb({type:'file', payload:req.body})
    res.send("OK");
})

app.put('/file', function (req, res) {
    console.log("PUT /file");
    updateInDb({type:'file', payload:req.body})
    res.send('OK');
})

app.delete('/file', function (req, res) {
    console.log("DELETE /file");
    removeFromDb({type:'file', payload:req.body})
    res.send('OK');
})

app.post('/filter', function (req, res) {
    console.log("POST /filter");
    appendDb({type:'filter', payload:req.body})
    res.send("OK");
})
```

Рисунок 5.2 — методи (ендпоінти) серверного API

Серверна частина розроблена згідно концепції SaaS (Software as a Service), тобто вона є мікро-сервісом, який надає повний доступ для серверної інтеграції іншими сервісами. Усі функції виконуються через HTTP запити:

- GET-запит на шлях «/» - повертає графічний інтерфейсу користувача (HTML файл)

- GET-запит на шлях «/static» - повертає необхідні скрипти та файли стилів для інтерфейсу користувача

- GET-запит на шлях «/api» - повертає стан динамічного реєстру користувачу;

- POST-запит на шлях «/open», з тілом запиту {link:<link>}, де «link» це посилання на файл в файловій системі серверу – відкриває вказаний файл на сервері. Підтримується запит лише із заголовком “Content-type”:”application/json”.

Шлях «/file» підтримує звернення з запитами типу:

- POST – для створення мета-інформації нового інформаційного ресурсу

- PUT – для редагування мета-інформації існуючого інформаційного ресурсу (обов’язкого вказати в запиті id інформаційного ресурсу).

- DELETE – для видалення мета-інформації існуючого інформаційного ресурсу (обов’язкого вказати в запиті id інформаційного ресурсу)

Шлях «/filter» підтримує звернення з запитами типу:

- POST – для створення нового фільтру

- PUT – для редагування існуючого фільтру (обов’язкого вказати в запиті id фільтру).

- DELETE – для видалення існуючого фільтру (обов’язкого вказати в запиті id фільтру).

Для реалізації клієнтської частини було використано фреймворк React, методологія використання якого зводиться до написання компонентів, які можно потім повторно використовувати. Проте через незначну кількість класів і невелику складність користувацького інтерфейсу – було розроблено лише головний компонент, в якому відбувається обробка і відображення головного вікна користувача та окремих екземплярів класів із доступними операціями над ними. Реалізація клієнтської частини може бути використана як частина серверної і водночас незалежною клієнтською платформою або десктопним застосунком при мінімальних змінах у коді (за допомогою фреймворку Electron).

Розроблена програмна система являє собою додаток для персонального комп’ютера або вебсайт (при віддаленому запуску) та запускається у середовищі

платформи NodeJS одразу для виконання. Для відображення застосунку необхідно мати встановлений веб-переглядач.

В даний момент, сервер працює на тестовому оточення і доступний усім в мережі Інтернет за наступним URL: https://mag2-toxcik3110.c9users.io/

Проте, маючи код серверної частини, можна запускати власні копії динамічних реєстрів із публічним, корпоративним або приватним доступом (зважуючи на те, в якій мережі знаходиться хост).

4.6. Графічний інтерфейс користувача

Після запуску користувач бачить перед собою головну форму (рисунок 6.1). Головна форма системи містить збережені згруповані та відсортовані посилання. Панель зверху дозволяє групувати за різними полями метадані, сортувати за різними полями та окремо реалізовано пошук, який фільтрує записи що не відповідають умовам пошуку. На панелі сортування доступна опція сортування у певному порядку або у зворотньому порядку.

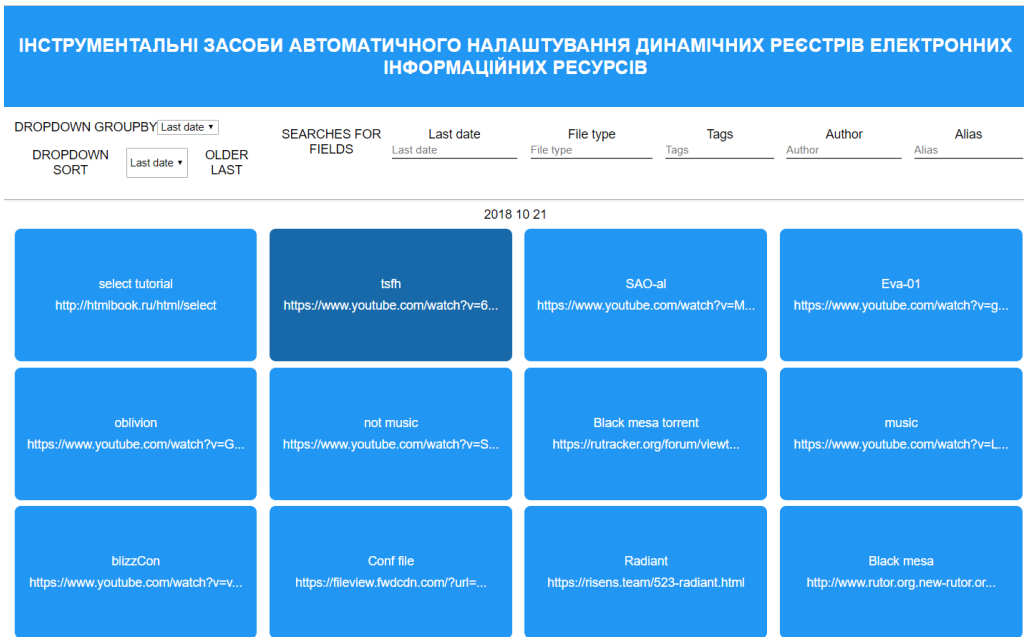

Рисунок 4.2 — Головна форма додатку

При натисканні на окремий запис посилання, відображається панель редагування з правої сторони (рисунок 6.2). Доступно для зміни лише частина метаданих, а саме: скорочена назва посилання, саме посилання, теги. Тип файлу

автоматично змінюється відповідно до зміни посилання і змінюється поведінка реакції на натискання кнопки “Link”, яка або відкриває посилання у веб-переглядачі або відкриває віддалений файл на сервері.

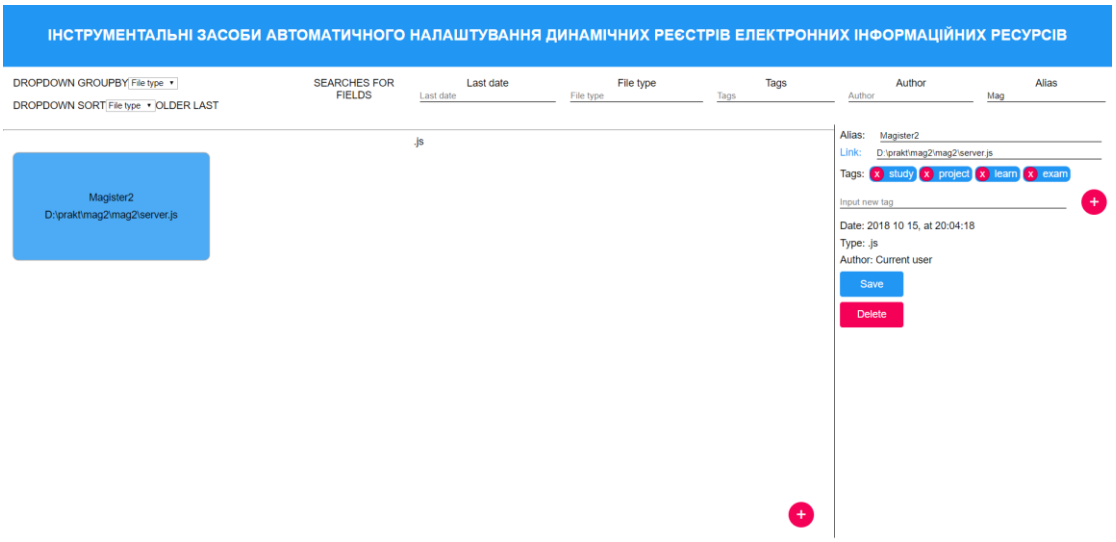

Рисунок 4.3 — Панель редагування посилання

При натисканні на кнопку «+», з’являється вікно створення посилання (рисунок 6.7).

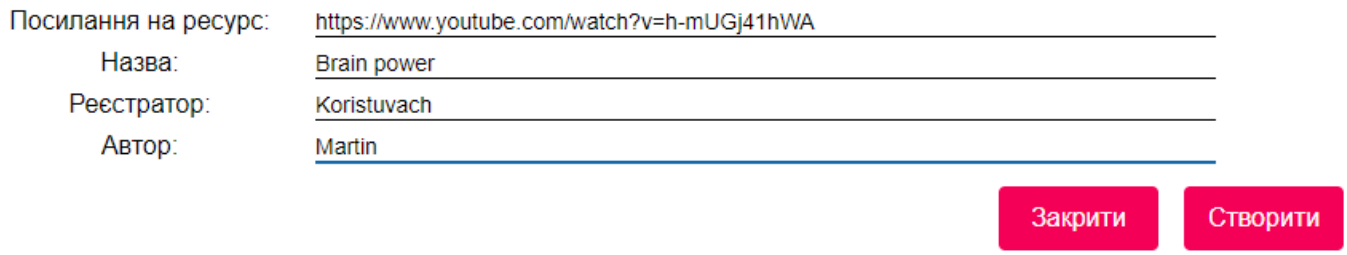

Рисунок 4.4 — Налаштування збереження в папку за замовчуванням

В ньому можна вказати скорочену назву нового посилання та саме посилання. Вся інша мета-інформація, а саме: час створення, автор, теги, тип посилання – буде згенеровано на основі самого посилання та відповідних характеристик оточення.

5. РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

Розділ має на меті проведення маркетингового аналізу стартап проекту задля визначення принципової можливості його ринкового впровадження та можливих напрямів реалізації цього впровадження. Проведення маркетингового аналізу передбачає виконання нижченаведених кроків.

Таблиця 5.1 – Опис ідеї проекту

Зміст ідеї	Напрями застосування	Вигоди для користувача
Створення сервісу, що містить посилання на електронні інформаційні ресурси та містить необхідні інструменти для фільтрації, сортування, налаштування доступу до різних груп посилань.	1. Збереження та налаштування електронних інформаційних ресурсів	Зручне зберігання та доступ до інформації
	2. Налаштування власних колекції інформаційних ресурсів.	Персоналізація колекції веде до відображення лише цільового контенту (те, що дійсно необхідно) для користувача.
	3. Доповнення і модерація існуючих колекцій інформаційних ресурсів.	Створення спільноти, заробіток на створенні контенту

Таблиця 5.2 - Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п	Техніко-економічн	(потенційні) товари/концепції конкурентів			S (сильна сторона)

/ п	і характеристики ідеї	Мій проект	Google chrome	Microsoft registry	W (слабка сторона)	N (нейтральна сторона)	
1.	Масштабованість проекту	Ще не масштабується	Масштабується	Не масштабується	Збільшує навантаження на систему, коштує грошей на кожне збільшення технічного парку, для впровадження ідеї потрібно змінювати архітектуру системи	Користувач може ніколи не використати цю особливість системи	Система стає більш стійкою, підтримує більше користувачів
2.	Спільний мережевий доступ до	Є спільний доступ	Немає спільного доступу	Немає спільного доступу	Потребує інтернет-з’єднання, навіть		Додає соціальні функції та способи монетиза

	динамічного реєстру				для локального користування, потребує механізмів синхронізації змін при одночасному користуванні одним й тим самим ресурсом декількома користувачами		ції проекту
3.	Авторизація та ролі користувачів	Є авторизація та розподіл ролей для	Є авторизація	Є авторизація і ролі доступу	Для повномірного користування всіма		Збільшує надійність та конфіденційність даних для

		кожного реєстру			функціями потрібно мати права адміністратора, можна втратити доступ до даних або управління проекту		приватних колекцій
4.	Користувацькі фільтри, сортування, пошук	Є користувацькі фільтри, авторизація, пошук, авто-фільтри	Є пошук та групування	Є пошук	Збільшують навантаження на базу даних при асинхронному транзакційному доступу до одного й		Збільшують комфортність роботи із системою, надають функції зручного доступу до кластеризації електронн

					того ж ресурсу		их інформаційних ресурсів
5.	Доступність ресурсу з будь-якого типу комп'ютера (мобільні пристрої, настільні ПК)	Так	Ні	Ні	В найгіршому випадку – підтримка декількох платформ та написання під кожну платформу окремого продукту	В кращому випадку – написання лише одного кросплатформного продукту	Користувач буде мати змогу використовувати систему будь-де, будь-коли

Таблиця 5.3. Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій

1.	Доступність з будь-якого пристрою	Веб-застосунок, що буде працювати по типу клієнт-серверної архітектури. Наприклад MEAN або LAMP-набір технологій або застосунок, що базується на веб-переглядачах (Electron, React-Native)	Технології наявні	Технології доступні
2.	Групування, сортування, фільтрація і пошук у динамічному реєстрі	Алгоритми кластеризації, машинне навчання, Tensor Flow	Технології наявні	Технології доступні
3.	Масштабованість проекту	Децентралізована система доступу до спільної бази даних або розподілені бази даних. Балансування навантаження на	Технології наявні	Технології доступні

		сервера, виділення необхідних потужностей за вимогою (Microsoft Azure, AWS, Google SRE)		
Буде використано технології серверного відображення вмісту динамічних реєстрів, алгоритми кластеризації для групування та фільтрації електронних інформаційних ресурсів				

Таблиця 5.4. Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	2
2	Загальний обсяг продаж, грн/ум.од	$90,3 млрд / рік
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Немає, окрім часу та зусиль
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	10%

Таблиця 5.5. Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1.	Збереження в пам’яті інформаційних ресурсів	Користувачі надзвичайно великих інформаційних потоків (користувачі мережі-Інтернет)	Активність користуванням різними інформаційними ресурсами (електронні бази знань, веб-сайти, електронні книги, мультимедіа), простота у користуванні сервісом, зручне доповнення платного функціоналу	- до продукції: Надійність роботи, 99% часу повинно бути доступно для користування (менше 1% часу на перезавантаження, оновлення, технічні роботи при поламці) - до компанії постачальника: Інформування про оновлення продукту, інтеграція з соціальними мережами, надання технічної підтримки та/або

				консультації щодо продукту

Таблиця 5.6. Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Тиск з боку держави	Фізичне знищення або вилучення серверів даних, блокування роботи персоналу.	Переміщення офісу та технічного оснащення за кордон.
2.	Маркетингова кампанія проти продукту	Використовуючи соціальні мережі, рекламні агенції, відгуки та статті на інформаційних веб-ресурсах зменшити довіру користувачів до продукту ті зменшення притоку нових користувачів.	Контр-реклама, використання відкритих виставок та онлайн-трансляцій з метою показу прозорості роботи системи для збільшення довіри користувачів до продукту.
3.	Некомпетентність кадрів	Можлива робота на конкурентів у внутрішніх кадрах,	Відповідно до проступку – від догани до

		невідповідальна робота над технічною реалізацією.	звільнення кадрів та пошук більш компетентних кадрів.
4.	Погано налагоджена технічна архітектура проекту	Не готовність продукту до навантажень від кількості реальних користувачів, недостатнє тестування певних компонентів продукту, можливий вміст комп’ютерних вірусів.	Своєчасне попередження користувачів продукту, правильний розрахунок часу розробки та розподіл зусиль на проект.

Таблиця 5.7. Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Поява інвесторів	Крупне інвестеційне товариство знаходить, що продукт можна використати для реклами або заробітку у	Створення бізнес-плану для задоволення потреб інвесторів, розробка додаткового

		перспективі та інвестує проект	функціоналу продукту
2.	Запрошення на участь у конференції	Можливість розказати про продукт на публічному просторі, що призведе до більш детального знайомства нових користувачів із існуючою системою	Після аналізу можливої аудиторії та позитивних результатів – склад та розробка доповіді та ораторів
3.	Влучне попадання у нишу	Комбінація існуючих технологій та потреб користувачів збігається і продукт починає користуватись популярністю	Пришвидшення випуску перших працюючих версій продукту для лімітованого кола користувачів, щоб раніше розпочати розкрутку продукту

Таблиця 5.8. Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії

		компанії, щоб бути конкурентоспроможною)
Чиста	Доволі легке копіювання продукту, тому основна боротьба буде в умовах «хто перший розробив та розкрутив»	Розробка схожого за функціоналом продукту та випуск його раніше за конкурентів або більш активна розкрутка.
Глобальний	Інтернет-технології налаштовані на вплив на аудиторію, що знаходиться по всьому світу.	Необхідність стежити за світовою ареною, найновішими рішеннями у світі.
Внутрішньогалузева	У більшості випадків, соціальні сервіси в мережі інтернет залишаються конкурувати у середовищі ІТ	Шукати способи вийти на міжгалузеву конкуренцію, пропонуючи користувачам більше способів взаємодії із продуктом
Між бажанням	Користувачі в більшості випадків користуються лише одним спеціалізованим на певній потребі сервісом у мережі Інтернет	Захоплювати ринок сегментами
Нецінова	Найбільша цінність – це кількість користувачів та їх утримання на	Після розробки програмного продукту, компанії необхідно

	використанні певного сервіс	концентруватись на маркетингу сервісу
Не марочна	При інтеграції соціального сервісу в інші продукти – важко оцінити хто відповідає за продукт	Зосереджуватись на розробці незалежного модуля, який буде легко впізнати користувачу

Таблиця 5.9. Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Google, Microsoft	Мають низькі капіталовкладення та доступ до ресурсів та технологій	Мають диференціацію цін на технічне обладнання	Можуть мати продуктову диференціацію	Товари замінники є завжди
Висновки:	Конкуренти завжди будуть намагатись використати більшу кількість ресурсів та більший вплив на соціальні	Є можливості виходу на ринок. Потенційні конкуренти існують. Вихід на ринок відбувається за строк від декількох	Постачальники не диктують умови	Користувачі дуже слідкують за якістю соціальних сервісів	Товари замінники існують, проте вони дуже слабко покривають потреби користувачів.

	медіа для збільшення кількості користувачів – і , як наслідок, домінації на ринку	місяців до півроку.			

Таблиця 5.10. Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1.	Швидкий вихід на ринок, розробка найбільш значущої функціональної частини продукту	Конкуренти націлені на інші види специфічних продуктів, тому будуть неконкурентоспроможні під час бурхливого розвитку маркетингу навіть напів-готового продукту
2.	Концентрація на розкрутці та піару продукту	При розробці мінімальної працюючої версії продукту першими і подальшій концентрації на маркетингу та доповненням сервісу – конкуренти повинні будуть зробити або теж саме (відстати на один етап розробки при аналогічному створенні мінімальної працюючої версії) або зосередитись на максимально повній версії програмного комплексу і втратити багато часу на якісний продукт, що

		призведе до складній конкуренції із вже розпіареним продуктом.

Таблиця 5.11. Порівняльний аналіз сильних та слабких сторін «назва проекту»

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з власним продуктом						
			-3	-2	-1	0	+1	+2	+3
1	Швидкий вихід на ринок	20							
2	Концентрація на розкрутці та піару продукту	10							

Таблиця 5.12. SWOT- аналіз стартап-проекту

Сильні сторони: Швидкий вихід на ринок Концентрація на розкрутці та піару продукту	Слабкі сторони: Низькі капіталовкладення Доступ до ресурсів та технологій
Можливості: Поява інвесторів Запрошення на участь у конференції Влучне попадання у нишу	Загрози: Тиск з боку держави Маркетингова кампанія проти продукту Некомпетентність кадрів Погано налагоджена технічна архітектура проекту

Таблиця 5.13. Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Карбування власної валюти, Проведення ICO та стабілізація курсу валюти, прив’язуючи її до стартап-проекту	50%	Від одного до кількох місяців із заборгованістю реалізації від одного року до декількох років

Таблиця 5.14. Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1.	Інвестори крипто-валют та децентралізованих анонімних мереж	70%	Максимальний попит при реалізованій системі вводу та виводу капіталу	Велика	Простий вхід
2.	Користувачі мережі Інтернет, котрі щодня стикаються з великою	50%	Користувачі ще не знають про свою	Немає	Простий вхід

	кількістю інформації		потребу у сервісі		
Які цільові групи обрано: 1 і 2					

Таблиця 5.15. Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку*
1.	Карбування власної валюти, Проведення ICO	Презентація проекту інвесторам, складання бізнес-плану, реалізація плану	Незалежне та стабільне фінансування, що дозволить стартапу існувати на декілька років більше ніж посередній стартап	Стратегія спеціалізації

Таблиця 5.16. Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*

1.	Проект є першопроходцем	Компанія буде шукати нових споживачів	Компанія буде копіювати деякі характеристики товару конкуренту, а саме: гнучкий пошук серед інформаційних ресурсів, персоналізація пошукової системи та фільтрів.	Стратегія лідеру

Таблиця 5.17. Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)

1.	Надійний та відказ-стійкий продукт з приємним та зрозумілим для нового користувача інтерфейсом	Стратегія спеціалізації	Можливість додати фінансування зі сторони інвесторів крипто-валют для забезпечення або потужного і швидкого старту або для довгого існування	ACCESS, IDACB, ABCA

Таблиця 5.18. Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1.	Запам’ятовування та фільтрація великої кількості інформаційних ресурсів	Синхронізація динамічних реєстрів електронних інформаційних ресурсів між усіма приладами користувача, здатність ділитись ресурсами з іншими користувачами	Кросплатформність рішення, синхронізація доступу та відображення

Таблиця 19. Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Запам’ятовування великої кількості інформаційних ресурсів, фільтрація, сортування та пошук серед них</td></tr>
<tr><td rowspan="5">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх/Тл/Е/Ор</td></tr>
<tr><td>1. Надійність, час роботи</td><td></td><td></td></tr>
<tr><td colspan="3">Якість: покриття тестами програмного коду</td></tr>
<tr><td colspan="3">Пакування – відсутнє</td></tr>
<tr><td colspan="3">Марка: Мй_Пркдт</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу – збільшений ліміт доступних динамічних реєстрів</td></tr>
<tr><td colspan="3">Після продажу – збільшення ліміту доступних динамічних реєстрів за окремі додаткові кошти</td></tr>
<tr><td colspan="4">Захист інтелектуальної власності</td></tr>
</table>

Таблиця 5.20. Визначення меж встановлення ціни

№ п/п	Рівень цін на товаризамінники	Рівень цін на товарианалоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1.	Великий	Безкоштовно	Середній	Товар безкоштовний, проте окремо можна придбати

				послуги на 5$ за транзакцію

Таблиця 5.21. Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1.	Цільові клієнти можуть закупити окремі послуги, що доповнюють основний проект, якщо будуть задоволені основні потреби.	Постачальник товару може виконувати функції фасування, сортування та складання товару на полицях веб-сайту дистрибутива	Продаж напряму від виробника	Децентралізований між користувачами інтернет-мережі

Таблиця 5.22. Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1.	Спілкуються через месенджери та соціальні мережі	Інтернет-мережі	1. Доступний сервіс для кожного. 2. Можна ділитись власними колекціями інформаційних ресурсів з усіма. 3. Перші, а значить краші	Звернути увагу на новий продукт, максимально розповсюдити и користування сервісом	Концепція рекламного звернення полягає у просуванні продукту серед експертів у інформаційному просторі

При аналізі стартап-моделі програмного продукту було визначено, що існує можливість комерціалізації проекту – наявний попит, ринок дуже динамічний, рентабельність присутня. Тому існують перспективи впровадження з огляду на потенційні групи клієнтів, бар’єри входження, стан конкуренції та можливу конкурентоспроможність проекту. Як альтернативу впровадження доцільно обрати проведення ICO для ринкової реалізації проекту. Подальша імплементація проекту є доцільною.

ВИСНОВКИ

Отже, при вирішенні завдання створення інструментальних ресурсів автоматизованого налаштування динамічних реєстрів електронних інформаційних ресурсів було розглянуто ознаки динамічності реєстрів, а саме:

- темпоральні ознаки;
- частотні ознаки;
- атрибутивні ознаки.

Було проаналізовано сучасні способи вирішення проблеми накопичення інформації; розглянуто основні способи кластеризації та класифікації інформації; продемонстровано існуючі рішення проблеми накопичення інформації у програмних середовищах і сервісах, а саме Google Web Search та Windows Registry.

В результаті дослідження було створено програмний комплекс, який складається із серверної та клієнтської частини. Клієнтська частина дозволяє користувачам динамічного реєстру створювати посилання на інформаційні ресурси, виконувати фільтрування, сортування та пошук серед реєстру електронних інформаційних ресурсів. Серверна частина програмного комплексу надає доступ до керування динамічним реєстром та виконана у вигляді мікро-сервісу, який можна використовувати для інтеграції з іншими сервісами та для розширення функціональних можливостей клієнтської частини, наприклад авторизації та адміністрування доступу до електронних інформаційних ресурсів та їх реєстрів.

Програмний комплекс має зручний та інтуїтивно зрозумілий інтерфейс.

Використані технології є типовими та за довгі роки використання визнаними індустрією як надійні та зручні у реалізації, а залучення до розробки останніх версій стеку технологій JavaScript (який дозволяє швидко розроблювати складні програмні комплекси не тільки для веб-застосунків, а і для платформно-залежних програм) та платформи NodeJS дозволяє створювати такі рішення серверної складової програмного забезпечення, надійним при надвеликих навантаженнях. Кросплатформність даного застосунку дозволить використовувати його на будь-яких операційних системах та пристроях, де встановлено веб-переглядач.

Було проведено дослідження стартап-моделі проекту та отримано результати аналізу, згідно яких проект може бути комерціалізованим через наявний попит, можливість конкурувати на ринку та перспективи впровадження і доцільність подальшої розробки.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бахтурина Т.А. «Проблеми стандартизации библиографического описания электронных ресурсов» // Науч. и техн. б-ки. — 2000. — № 7. — С. 16-21.
2. Arpaci-Dusseau, Remzi H .; Arpaci-Dusseau, Andrea C. (2014), впровадження файлової системи (PDF), Arpaci-Dusseau Books
3. Arpaci-Dusseau, Remzi H.; Arpaci-Dusseau, Andrea C. (2014), Sun's Network File System (PDF), Arpaci-Dusseau Books
4. Амір Яір. "Операційні системи 600.418 Файлова система". Кафедра комп'ютерних наук Університету Джона Хопкінса. Отримано 31 липня 2016 року.
5. Корпорація IBM. "Компонентна структура логічної файлової системи". IBM Knowledge Center. Отримано 31 липня 2016 року.
6. Р. С. Дейлі; П. Г. Нейман (1965). Загальноприйнята файлова система для вторинного зберігання. Осінь Спільна комп'ютерна конференція. AFIPS. Стор 213-229.
7. Мохан, І. Чандра (2013 р.). Операційні системи. Делі: PHI Learning Pvt. ТОВ стр. 166. ISBN 9788120347267.
8. https://www.researchgate.net/publication/234789457_KSAM_A_B_-tree-based_keyed_sequential-access_method
9. https://en.wikipedia.org/wiki/TheFreeDictionary.com
10. https://www.millforbusiness.com/how-many-websites-are-there/
11. https://news.netcraft.com/archives/category/web-server-survey/
12. http://www.internetlivestats.com/total-number-of-websites/
13. https://developers.google.com/web/fundamentals/design-and-ui/responsive/
14. Тріон, Роберт С. (1939). Кластерний аналіз: кореляційний профіль і ортометричний (факторний) аналіз для виділення єдностей у свідомості та особистості. Брати Едвардс.

15. Скуллі, Д. (2010). Кластеризація k-means у веб-масштабах. Проц. 19-а WWW.
16. Хуан, З. (1998). "Розширення алгоритму k-means для кластеризації великих наборів даних з категоріальними значеннями". Видобування даних та виявлення знань. 2: 283-304.
17. Р. Н. Г. і Д. Хан. "Ефективний і ефективний метод кластеризації для виявлення просторових даних". В: Праці 20-ї конференції VLDB, стор 144-155, Сантьяго, Чилі, 1994.
18. Тянь Чжан, Рагху Рамакрішнан, Мирон Лівні. "Ефективний метод кластеризації даних для дуже великих баз даних". В: Proc. Int'l Conf. «Управління даними», ACM SIGMOD, стор. 103-114.
19. Крігель, Ганс-Петро; Kröger, Peer; Зімек, Артур (липень 2012 р.). "Кластеризація підпрограм". Міжрегіональні огляди Wiley: виявлення даних та знання. 2 (4): 351-364.
20. Агравал, Р .; Gehrke, J .; Гунопулос, Д .; Раггаван, П. (2005). "Автоматичне підгрупування кластеризації високоякісних даних". Видобування даних та виявлення знань
21. Карин Кайлінг, Ганс-Петро Крігель та П'єр Крегер. Кластеризація з підключенням до великогабаритних даних, що підключаються до щільності. В: Proc. SIAM Int. Конф на Data Mining (SDM'04), стор. 246-257, 2004.
22. Achtert, E .; Böhm, C .; Крігел, Г.-П .; Kröger, P .; Мюллер-Горман, I .; Зімек А. (2006). "Пошук ієрархій кластерів підпрограм". LNCS: Відкриття знань у базах даних: PKDD 2006. Лекційні замітки в області комп'ютерних наук. 4213: 446-453. до: 10.1007 / 11871637_42. ISBN 978-3-540-45374-1.
23. Achtert, E .; Böhm, C .; Крігел, Х. П .; Kröger, P .; Мюллер-Горман, I .; Зімек А. (2007). "Виявлення та візуалізація кластерних ієрархій підпрограм". LNCS: досягнення в базах даних: поняття, системи та програми. Лекційні замітки в області комп'ютерних наук. 4443: 152-163. до: 10.1007 / 978-3-540-71703-4_15. ISBN 978-3-540-71702-7.

24. Achtert, E .; Böhm, C .; Kröger, P .; Зімек А. (2006). "Гірські ієрархії кореляційних кластерів". Проц. 18-а Міжнародна конференція з управління науковими та статистичними даними (SSDBM): 119-128. до: 10.1109 / SSDBM.2006.35. ISBN 0-7695-2590-3.
25. Böhm, C .; Кайлінг, К .; Kröger, P .; Зімек А. (2004). Msgstr "Обчислювальні кластери пов'язаних об'єктів". Праця міжнародної конференції ACM SIGMOD 2004 року "Управління даними" - SIGMOD'04. p. 455. до: 10.1145 / 1007568.1007620. ISBN 1581138598.
26. Achtert, E .; Бом, C .; Крігел, Х. П .; Kröger, P .; Зімек А. (2007). "Дослідження складних зв'язків кореляційних кластерів". XI Міжнародна конференція з управління науковими та статистичними даними (SSDBM 2007). p. 7. doi: 10.1109 / SSDBM.2007.21. ISBN 0-7695-2868-6.
27. Мале, Марина (2003). "Порівняння кластерій за варіаціями інформації". Теорія навчання та машини ядра. Лекційні замітки в області комп'ютерних наук. 2777: 173-187. Дой: 10.1007 / 978-3-540-45167-9_14. ISBN 978-3-540-40720-1.
28. Красков, Олександр; Stögbauer, Harald; Анджеяк, Ральф Г .; Грассбергер, Петро (1 грудня 2003 р.). "Ієрархічна кластеризація на основі взаємної інформації".
29. Ауфарт, Б. (18-23 липня 2010 р.). "Кластеризація за допомогою генетичного алгоритму з оператором навмисного мутації". Wcci Cec. IEEE.
30. Фрей, Б. Дж.; Dueck, D. (2007). "Кластеризація шляхом передачі повідомлень між точками даних". Наука. 315 (5814): 972-976.
31. Пфіцнер, Дарій; Лейбрандт, Річард; Держави, Давид (2009). "Характеристика та оцінка заходів подібності для пар кластерингу". Знання та інформаційні системи. Спрінгер 19: 361-394.
32. Фельдман, Ронен; Сенгер, Джеймс (2007-01-01). Довідник з розробки текстових текстів: вдосконалені підходи при аналізі неструктурованих даних. Cambridge Univ. Прес ISBN 0521836573.

33. Вайс, Шолом М.; Індурхья, Нітин; Чжан, Тонг; Дамера, Фред Дж. (2005). Текстовий видобуток: передбачувані методи аналізу неструктурованої інформації. Спрінгер ISBN 0387954333. OCLC 803401334.

ДОДАТОК А

Публікації

Інструментальні засоби автоматизованого налаштування динамічних реєстрів електронних інформаційних ресурсів

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ3188_18М

Аркушів 4

2018

Проблеми автоматичного налаштування динамічних реєстрів електронних інформаційних ресурсів // Карпенко С.Г., Чайка А.Ю. // Сучасні проблеми наукового забезпечення енергетики: Матеріали XVI Міжнародної науково-практичної конференції аспірантів, магістрів і студентів, м. Київ, 24-27 квітня 2018 р. У К.: КПІ ім. Ігоря Сікорського», 2018. – С. 148.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

СУЧАСНІ ПРОБЛЕМИ НАУКОВОГО ЗАБЕЗПЕЧЕННЯ ЕНЕРГЕТИКИ

Матеріали XVI Міжнародної науково-практичної конференції аспірантів, магістрантів і студентів м. Київ, 24-27 квітня 2018 року,

ТОМ 2

НТУУ "КПІ" 1931 ТЕФ

Українське ядерне товариство

ЕНЕРГОАТОМ ENERGOATOM

Київ- 2018

УДК 004.457

Магістрант 5 курсу, гр. ТІ-71мп Чайка А.Ю.
Доц., к.ф-м.н. Карпенко С.Г.

Проблеми автоматичного налаштування динамічних реєстрів електронних інформаційних ресурсів

Дедалі ширше використання електронних документів [1] виявляє серйозні проблеми практичної реалізації використання реєстрів електронних інформаційних ресурсів [2-4], пов'язаних із створенням і використанням електронних ресурсів, із оптимізацією системи класифікації категорій реєстрів, яка б забезпечувала адекватну і зрозумілу передачу змісту документів з метою їх ідентифікації, упорядкованого зберігання та швидкого пошуку в електронних ресурсах, передбачала би пошук за різними критеріями (відповідно до специфіки електронних пошукових систем). Система має вільно оперувати електронними документами незалежно від зміни їх типів, видів, тощо.

Основні проблеми, що повстають при реалізації реєстрів електронних ресурсів, є методи класифікації електронних інформаційних ресурсів, методи створення та налаштування реєстрів, проблеми збереження ресурсів, забезпечення динамічного характеру реєстрів.

У процесі розробки методичних засад каталогізації електронних ресурсів особливої уваги набуває дослідження їх топології, що є необхідним для виділення груп електронних ресурсів за суттєвими типовими ознаками та визначення технологічних процедур бібліотечного опрацювання різних видів електронних ресурсів. До питань типології слід додати питання визначення статусу документа та розробки методик опису електронного ресурсу як оригіналу, а також відтворення будь-яких об'єктів.

Для збереження електронних документів, враховуючи їх великий обсяг (крім застосування серверів локальних мереж), повстають проблеми використання хмарних технологій, програмне забезпечення віддаленого доступу для збереження електронної інформації, для його редагування, візуального відображення, зчитування (браузери, редактори, спеціалізовані програми для перегляду та друкування).

Важливими аспектами використання реєстрів електронних інформаційних ресурсів є програмне забезпечення динамічного характеру реєстру, можливості зміни структури реєстрів як в процесі його застосування, так й при зміні змістового вмісту реєстрів. Крім того, мають бути реалізовані різні рівні доступу до електронних інформаційних ресурсів для різних категорій користувачів.

Перелік посилань:

1. Баркова О.В. Питання організації фонду онлайнових документів електронної бібліотеки // Реєстрація, зберігання, обробка даних. – 2002. – Т. 4, N 2. – С. 85-95.

2. Про електронні документи та електронний документообіг: Закон України від 22 травня 2002 р. № 851-IV // Урядовий кур'єр – 2003. – 2 липня. - №119. – С.1-6.
3. Поняття інформаційних ресурсів. Режим доступу: http://studies.in.ua/inform-pravo-shporu/518-ponyattya-nformacynih-resursv.html.
4. Про інформацію: Закон України від 2 жовтня 1992 року № 2567-XII // Відомості Верховної Ради України. – 1992. – № 48. – С.650.

ДОДАТОК Б

Акт впровадження

Інструментальні засоби автоматизованого налаштування динамічних реєстрів електронних інформаційних ресурсів

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ ТІ3188_18М

Аркушів 2

2018